Nokia AirScale Configuration Guide

Configuring Base Stations for RAN Monitor Integration

Table of Contents

Overview

To enable RAN Monitor to collect performance metrics, alarms, and configuration data from Nokia AirScale base stations, you must configure the base station to report data to the RAN Monitor system. This is accomplished through the Nokia Web Element Manager (WebLM) interface.

This guide walks through the process of configuring the Performance Measurement Common Administration (PMCADM) managed object, which controls how the base station sends performance data to external systems.

Who Should Use This Guide

Important: All Nokia AirScale base station configuration is **performed by Omnitouch** as part of the initial deployment and ongoing support. This guide is provided for:

- **Advanced users** who want to understand the base station configuration
- **Self-managed deployments** where customers configure their own base stations
- **Troubleshooting** and understanding the current configuration
- **Additional base station onboarding** in self-managed environments

If you are an Omnitouch-managed customer, contact Omnitouch support for base station configuration and onboarding.

For understanding the metrics being collected, see Nokia Counter Reference.
For system configuration, see Runtime Configuration Guide.

Prerequisites

Before configuring the base station, ensure you have:

- **Network Access** - Connectivity to the base station management interface
- **Administrator Credentials** - Username and password with configuration privileges
- **RAN Monitor Details** - IP address and port where RAN Monitor is listening
- **Supported Software** - Compatible Nokia AirScale base station software version

Required Information:

Parameter	Value	Example
RAN Monitor IP Address	IP where RAN Monitor runs	10.179.2.139
RAN Monitor Port	Collection port (default: 9076)	9076
Collection Interval	How often to send metrics	60 seconds

Accessing the WebLM Interface

Step 1: Open the Web Element Manager

1. Open a web browser
2. Navigate to the base station management interface:

   ```
   http://<base-station-ip>/
   ```

   or

   ```
   https://<base-station-ip>/
   ```

3. Log in with your administrator credentials

Step 2: Navigate to Configuration Management

Once logged in:

1. Click **Configuration** in the top menu bar
2. Select **Configuration Management** from the dropdown
3. Click the **Parameter Editor** tab

You should now see the configuration tree in the left panel and parameter editor in the main window.

Configuring Performance Monitoring

Step 1: Locate the PMCADM Managed Object

In the left navigation panel (Objects tree):

1. Expand **Current BTS configuration**
2. Expand **CURRENT_BTS_CONF-1**
3. Expand **MRBTS-X** (where X is your base station ID)
4. Expand **MNL-1** (Management Link)
5. Expand **MNLENT-1** (Management Link Entity)
6. Click on **PMCADM-1** (Performance Measurement Common Administration)

The parameter editor will display the PMCADM-1 configuration parameters.

Step 2: Configure Real-Time Performance Monitoring

Scroll down to the **Structure 1** section, which contains the Real-Time Performance Monitoring Collection Entity settings. Configure the following parameters:

Required Parameters:

Parameter	Description	Recommended Value
Certificate type for TLS authentication	Security certificate type	RSA (if TLS enabled)
Real time performance monitoring collection entity host	IP address of RAN Monitor	Your RAN Monitor IP (e.g., 10.179.2.139)
Real Time Performance Monitoring Collection Entity Port Number	Port where RAN Monitor listens	9076 (default)
Real Time Performance Monitoring Collection Interval	How often to send metrics	60s (adjust based on requirements)
Enable TLS	Use encrypted connection	false (for initial setup)

Optional Parameters:

Parameter	Description	Default Value
SDL maximum upload file number	Max concurrent uploads	1
SDL Nonce	Unique identifier for security	12345678
Enable zero value counters suppression	Suppress counters with value 0	false (recommended to keep all data)

Step 3: Save and Activate Configuration

After configuring all parameters:

1. **Review your changes** - Verify all IP addresses and ports are correct
2. **Create a configuration plan:**
    - Click **Create Plan** button at the top
    - The system will validate your changes
    - Note the Plan ID provided
3. **Validate the plan:**
    - Click **Validate Plan** button
    - Enter the Plan ID
    - Wait for validation to complete
    - Address any validation errors
4. **Activate the configuration:**
    - Click **Activate Plan** button
    - Enter the Plan ID
    - Confirm activation
    - The base station will apply the new configuration

Alternative: XML Configuration

For advanced users or automated deployments, the PMCADM configuration can be applied using XML. This is the configuration snippet that corresponds to the manual configuration above:

```
<managedObject class="com.nokia.srbts.mnl:PMCADM" distName="MRBTS-132/MNL-1/MNLENT-1/PMCADM-1" version="MNL25R1_2420_110" operation="create">
  <p name="act3gppXmlEnrichment">false</p>
  <p name="reportingIntervalPm">5min</p>
  <p name="sdlMaxUploadFileNumber">1</p>
  <p name="sdlPrimaryDestIp">10.179.2.139</p>
  <list name="rTPmCollEntity">
    <item>
      <p name="certTypeForTlsAuth">RSA</p>
      <p name="rTPmCollEntityHost">10.179.2.139</p>
      <p name="rTPmCollEntityPortNum">9076</p>
      <p name="rTPmCollInterval">60s</p>
      <p name="tlsEnabled">false</p>
    </item>
  </list>
</managedObject>
```

Key Parameters in XML:

- `rTPmCollEntityHost` - Set to your RAN Monitor IP address
- `rTPmCollEntityPortNum` - Set to 9076 (default webhook port)
- `rTPmCollInterval` - Collection interval (60s recommended)
- `tlsEnabled` - Set to false for initial setup
- `sdlPrimaryDestIp` - Set to your RAN Monitor IP address

Note: Replace `10.179.2.139` with your actual RAN Monitor IP address and adjust `MRBTS-132` to match your base station ID.

Configuration Parameters Reference

PMCADM-1 Object Overview

The PMCADM (Performance Measurement Common Administration) managed object controls how performance data is collected and reported from the base station.

Key Responsibilities:

- Configure real-time performance monitoring destinations
- Set collection intervals for metrics reporting
- Control data format and transmission parameters
- Manage security settings for data transmission

Real-Time Performance Monitoring Collection Entity

This sub-structure defines where and how the base station sends real-time performance metrics.

certTypeForTlsAuth - Certificate Type for TLS Authentication

- **Type:** Enumeration (RSA, DSA, ECDSA)
- **Purpose:** Specifies the certificate type when TLS is enabled
- **Default:** RSA
- **Usage:** Only relevant when tlsEnabled = true

rTpmCollEntityHost - Collection Entity Host

- **Type:** IP Address (IPv4 or IPv6)
- **Purpose:** Destination IP address for performance metrics
- **Required:** Yes
- **Example:** 10.179.2.139
- **Notes:** Must be reachable from base station management network

rTpmCollEntityPortNum - Collection Entity Port Number

- **Type:** Integer (1-65535)
- **Purpose:** TCP port where collection system listens
- **Default:** 9076
- **Notes:** Must match RAN Monitor configuration

rTpmCollInterval - Collection Interval

- **Type:** Time (seconds)
- **Purpose:** Frequency of performance data transmission
- **Options:** 15s, 30s, 60s, 300s, 900s, 1800s
- **Default:** 60s
- **Recommendation:** 60s for standard monitoring, 15s for detailed troubleshooting

tlsEnabled - Enable TLS

- **Type:** Boolean (true/false)
- **Purpose:** Encrypt performance data in transit
- **Default:** false
- **Notes:** Requires valid certificates on both sides if enabled

sdlMaxUploadFileNumber - SDL Maximum Upload File Number

- **Type:** Integer
- **Purpose:** Maximum number of concurrent file uploads
- **Default:** 1
- **Notes:** Increase for high-volume environments

sdlNonce - SDL Nonce

- **Type:** String (8 digits)
- **Purpose:** Unique identifier for SDL protocol security
- **Default:** 12345678
- **Notes:** Can be changed for security purposes

suppressZeroValueCount - Suppress Zero Value Counters

- **Type:** Boolean (true/false)
- **Purpose:** Omit counters with zero values from reports
- **Default:** false
- **Recommendation:** Keep false to maintain complete data for trending

---

Verification

After activating the configuration, verify the base station is successfully sending data to RAN Monitor.

Check RAN Monitor Web UI

1. Open the RAN Monitor Web UI: `http://<ran-monitor-ip>:4000/`
2. Navigate to **eNodeB Status** page
3. Locate your base station in the device list
4. Verify the status shows **Connected** (green indicator)
5. Check that **Session** shows **Active**

Expected Status:

- **Status:** Connected (green)
- **Session:** Active
- **Address:** Matches base station IP
- **Actions:** All buttons enabled

If Status Shows Pending:

The device is attempting to register but hasn't completed authentication.

Possible causes:

- Manager ID and registration key mismatch
- RAN Monitor not configured to accept this device
- Network connectivity issues

If Status Shows Connection Error:

The device cannot reach RAN Monitor.

Possible causes:

- Incorrect IP address in PMCADM configuration
- Network routing issues
- Firewall blocking port 8080
- RAN Monitor service not running

Check Data Collection

View InfluxDB Status:

1. In RAN Monitor Web UI, navigate to **InfluxDB Status**
2. Verify data points are increasing
3. Check **Performance Metrics** count is growing
4. Confirm **Last Update** timestamp is recent

Expected Metrics:

- **Performance Metrics:** Count increasing regularly
- **Configuration:** Data points present
- **Alarms:** May be 0 if no faults active

Check Data Retention:

1. Navigate to **Data Retention Policy** page
2. Locate your base station
3. Verify **Performance Metrics**, **Configuration**, and **Alarms** counts

---

Troubleshooting

Base Station Not Appearing in RAN Monitor

Symptom: Device doesn't show up in eNodeB Status page

Solutions:

1. **Verify network connectivity:**

```
ping <base-station-ip>
```

2. **Check RAN Monitor configuration:**
    - Ensure device is added to `config/runtime.exs`
    - Verify IP address matches base station
    - Confirm credentials are correct
3. **Review RAN Monitor logs:**
    - Navigate to **Live Logs** page
    - Filter for error messages
    - Look for connection attempts from the base station
4. **Verify base station configuration:**
    - Re-check PMCADM-1 settings in WebLM
    - Confirm RAN Monitor IP address is correct
    - Ensure port 9076 is specified

Device Shows "Pending" Status

Symptom: Device appears but status is yellow "Pending"

Solutions:

1. **Check manager registration:**
    - Verify manager ID in RAN Monitor matches base station expectation
    - Confirm registration keys are properly configured
2. **Review authentication:**
    - Check credentials in runtime.exs
    - Ensure username/password match base station settings
3. **Wait for registration cycle:**
    - Registration may take 30-60 seconds
    - Refresh the page after waiting

Connection Errors

Symptom: "Network error: enetunreach" or similar

Solutions:

1. **Verify network path:**
   - Test connectivity from base station to RAN Monitor
   - Check routing tables
   - Verify VLANs/subnets are correctly configured
2. **Check firewall rules:**
   - Ensure port 9076 is open (for real-time performance data)
   - Ensure port 8080 is open (for SOAP API communication)
   - Verify no ACLs blocking traffic
   - Check iptables rules on RAN Monitor server
3. **Verify RAN Monitor is listening:**

```
# Check both SOAP API and webhook endpoints
netstat -tlnp | grep -E '8080|9076'
```

No Metrics in InfluxDB

Symptom: Device is connected but no data in database

Solutions:

1. **Verify collection interval:**
   - Check PMCADM-1 rTpmCollInterval setting
   - Wait at least one full interval period
   - Refresh InfluxDB Status page
2. **Check InfluxDB connection:**
   - Navigate to InfluxDB Status page

- Verify "Connected" indicator is green
- Confirm bucket name is correct

3. **Review RAN Monitor logs:**
   - Look for InfluxDB write errors
   - Check for data parsing issues
   - Verify API token has write permissions

TLS/Certificate Issues

Symptom: Connection fails when TLS is enabled

Solutions:

1. **Verify certificates are installed:**
   - Check base station has valid certificate
   - Ensure RAN Monitor has corresponding CA certificate
2. **Try without TLS first:**
   - Set tlsEnabled = false
   - Verify basic connectivity works
   - Re-enable TLS after confirming functionality
3. **Check certificate validity:**
   - Verify certificates are not expired
   - Confirm certificate subject names match
   - Check certificate chain is complete

Additional Resources

Related Documentation

- Operations Guide - Complete RAN Monitor operations documentation
- Runtime Configuration Guide - RAN Monitor configuration reference
- Nokia Counter Reference - Performance counter definitions
- Grafana Integration - Building dashboards with collected metrics
- API Endpoints - REST API reference for device management
- Data Retention Policy - Managing stored performance data

Configuration Files

- **config/runtime.exs** - RAN Monitor device configuration

Support

For issues not covered in this guide:

1. Review RAN Monitor application logs
2. Check Nokia base station documentation for your software version
3. Verify network infrastructure configuration
4. Consult with network operations team

Alarm Management & Escalation

Fault Handling, Severity Levels, and Operational Response

Guide to managing alarms, investigating faults, and escalating issues

---

Table of Contents

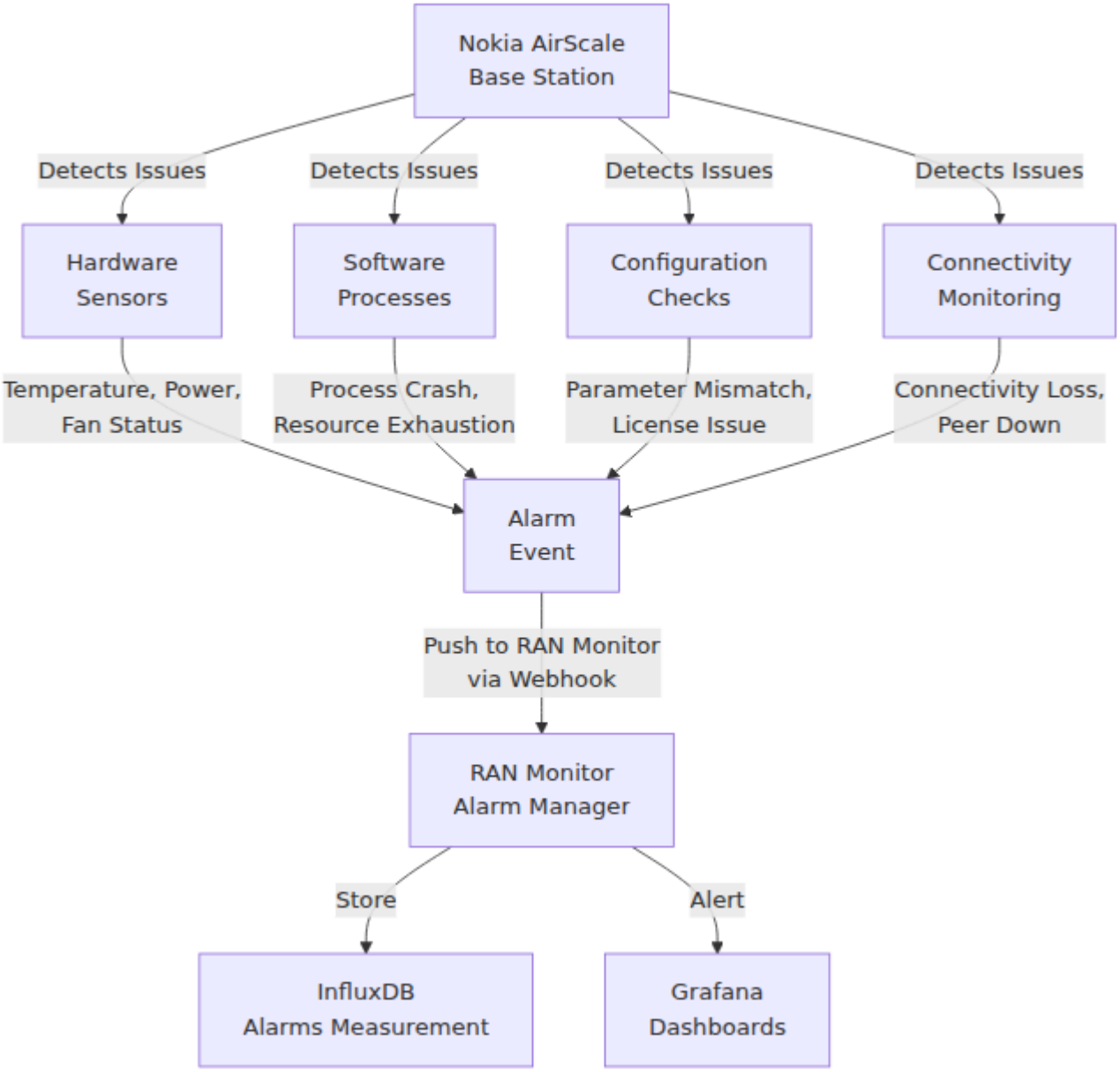
Overview

Alarms (also called "faults") represent detected problems or anomalies on Nokia AirScale base stations. RAN Monitor continuously monitors active alarms and tracks their lifecycle from generation through resolution.

Example Alarm Dashboard:

Example showing 4G Status with alarm overview table displaying alarm status (active/cleared), severity levels (critical/warning), timestamps, and alarm descriptions for optical interface failures.

Alarm Sources



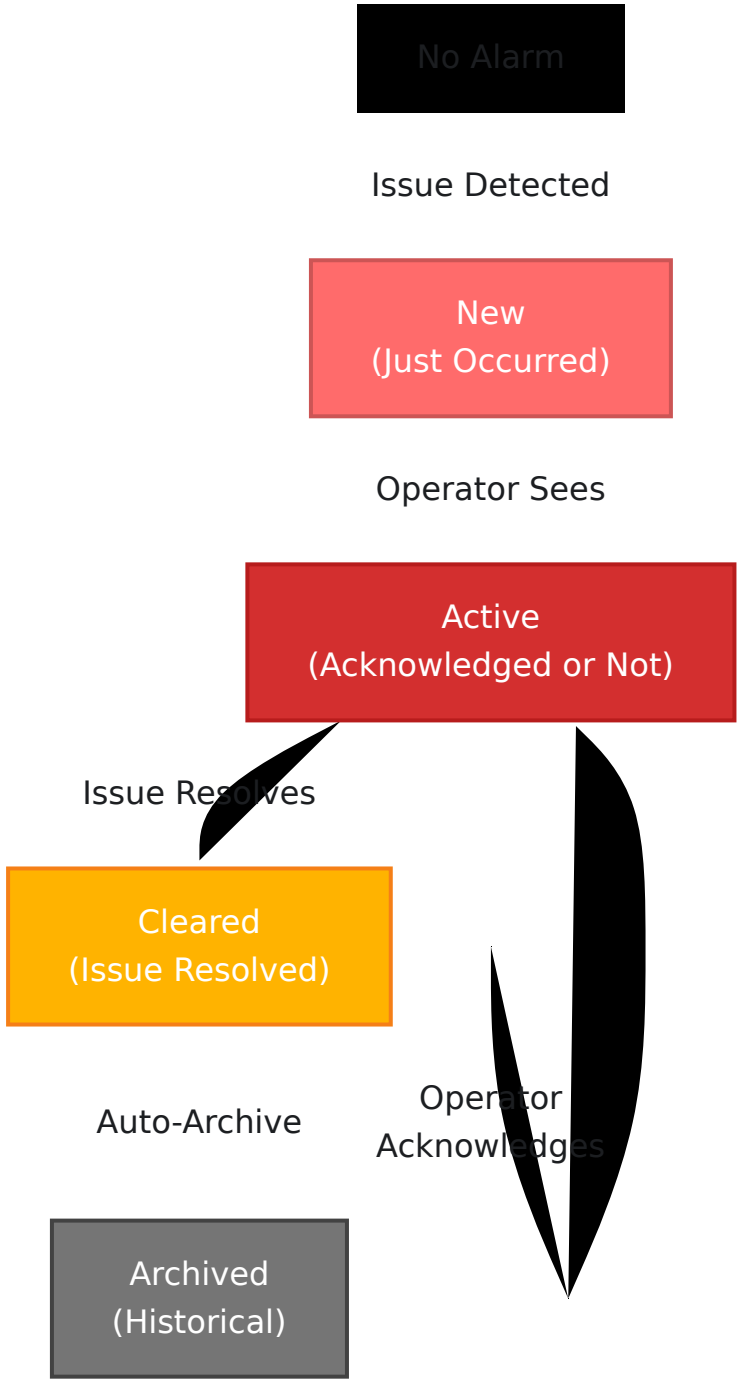
Key Alarm Attributes

Every alarm contains:

Attribute	Example	Purpose
Alarm ID	a1b2c3d4-e5f6-...	Unique identifier
Severity	Critical, Major, Minor	Priority level
Probable Cause	"Cell Unavailable"	Root cause category
Specific Problem	"S1 Connection Lost"	Detailed issue
Affected System	/BSC-1/BTS-23/Cell-A	What's impacted (DN)
Event Time	2025-12-10 14:23:45	When detected
Status	Active / Cleared	Current state

Alarm Lifecycle

State Transitions



Example Alarm Timeline

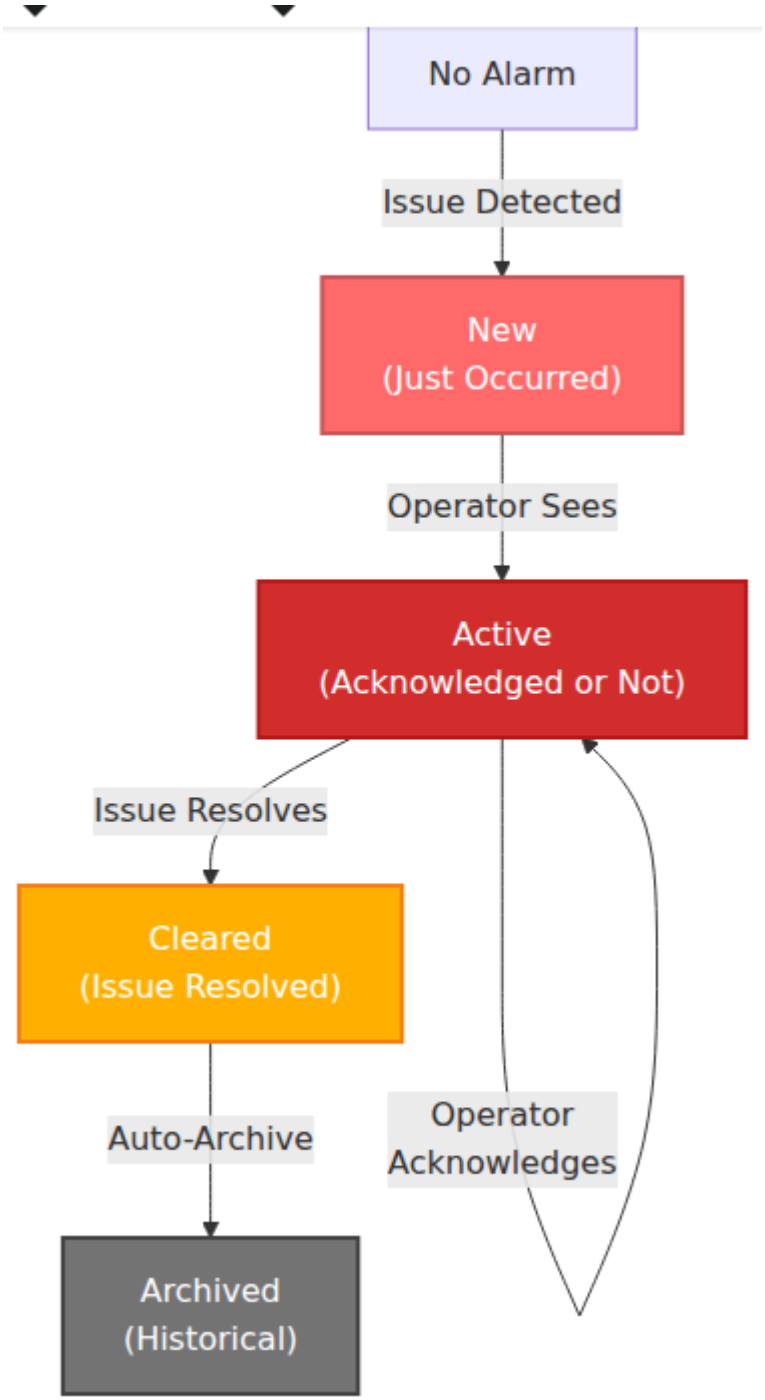
```
14:23:45 UTC - Issue Occurs
            └─> Base station detects connectivity loss
            └─> Generates alarm: "S1 Connection Lost" (Critical)

14:23:47 UTC - Alarm Pushed to RAN Monitor
            └─> NE3S webhook notification received
            └─> Stored in InfluxDB
            └─> Alert rule triggered

14:23:50 UTC - Notification Sent
            └─> Grafana alert fires
            └─> Slack message to NOC team
            └─> PagerDuty incident created

14:24:15 UTC - Operator Acknowledges
            └─> NOC team marks as acknowledged
            └─> Duration tracking begins

14:28:35 UTC - Issue Auto-Resolves
            └─> Connectivity restored
            └─> Base station clears alarm
            └─> RAN Monitor records "Cleared"

14:28:36 UTC - Alarm Closed
            └─> Duration recorded: 5 minutes 51 seconds
            └─> Tracked for SLA reporting
            └─> Archived after 30 days
```

Severity Levels

RAN Monitor tracks five severity levels, each with different operational impact and escalation requirements:

Critical Severity

Definition: Service-impacting, requires immediate action



Examples:

- Device completely unreachable (connectivity loss)
- All cells down (baseband failure)
- Control plane interface down (S1 lost)
- Complete data forwarding failure
- Base station not responding to management

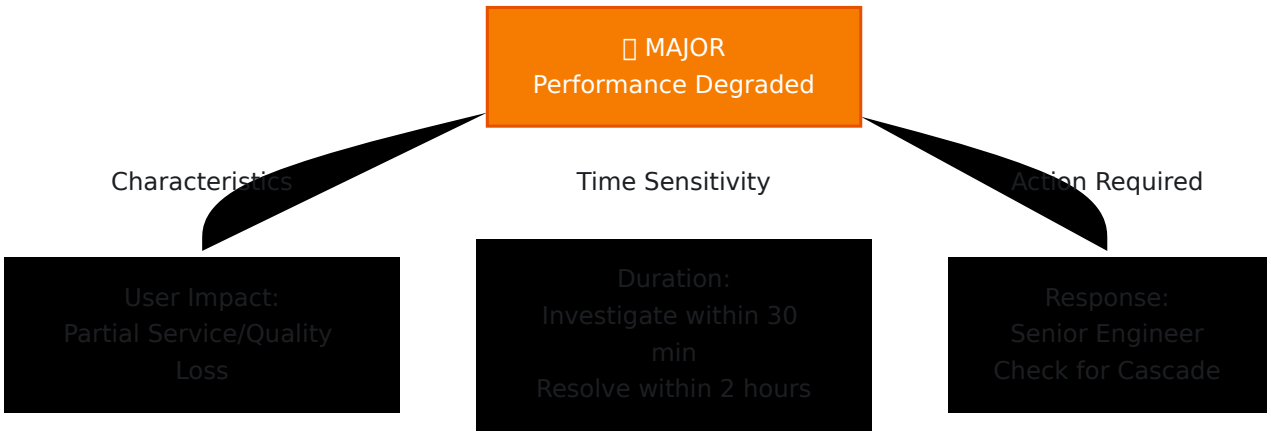
Escalation:

- Notify on-call engineer immediately (phone call)
- Create incident in incident management system
- Update status page
- Inform management if SLA affected

Response SLA: < 15 minutes

Major Severity

Definition: Degraded performance, requires urgent investigation



Examples:

- Cell availability < 95% for > 15 minutes
- Handover success rate < 95%
- DL/UL resource blocked (> 95% utilization sustained)
- RLC retransmission rate > 5%
- Multiple cells showing poor quality
- Link degradation (E1/T1 errors increasing)

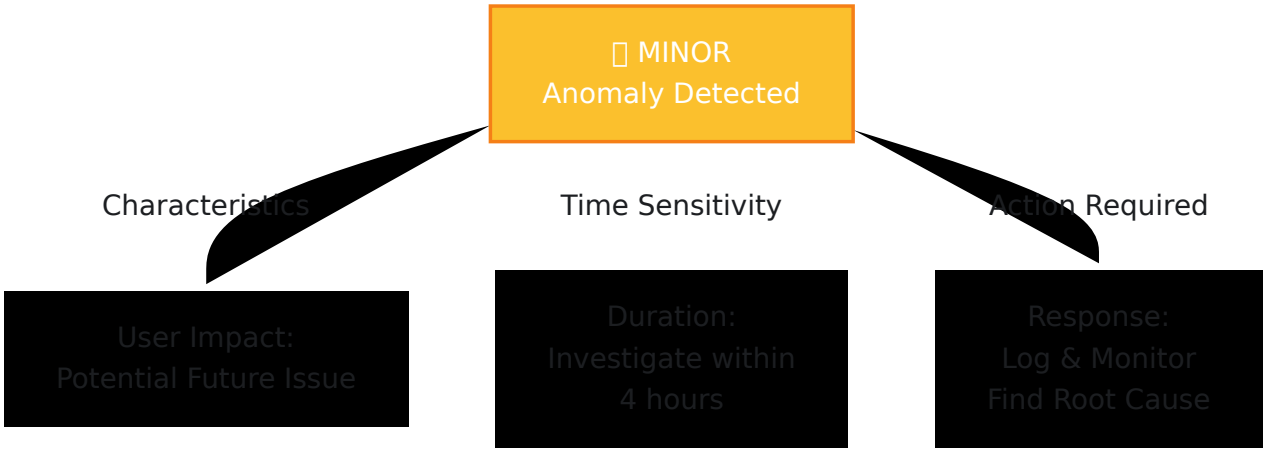
Escalation:

- Notify NOC team + senior engineer
- Create incident in incident management
- Page engineering team if still open after 30 minutes
- Check for cascading issues to other cells/sites

Response SLA: < 30 minutes investigation

Minor Severity

Definition: Degradation, track and investigate



Examples:

- Cell availability 95-98% (trending down)
- High temperature warning on amplifier
- License capacity approaching limit
- Configuration consistency issues
- Slow performance on management interface
- Intermittent alarms (< 5 occurrences/hour)

Escalation:

- Log in dashboard for awareness
- Assign to engineering for investigation
- Schedule for next maintenance window if needed
- Create ticket for trend analysis

Response SLA: Same day review

Warning Severity

Definition: Informational, monitor for trends

Examples:

- Cell availability > 98% but trending down
- Temperature/power in normal range but elevated
- Resources at 60-70% utilization
- Configuration mismatch (non-critical parameters)
- First occurrence of new fault type

Escalation:

- Dashboard visibility only
- No automatic notifications
- Manual review on cadence

Cleared

Definition: Previously active alarm now resolved

Purpose:

- Documents that issue has been resolved
- Tracks mean time to repair (MTTR)
- Enables SLA compliance reporting
- Identifies repeat problems

Alarm Categories

Connectivity Alarms

```
Category: External Connectivity
Probable Causes:
  - S1 Connection Lost       → MME/SGW unreachable
  - Backhaul Link Down       → IP transport failure
  - USIM Interface Error     → HSS connectivity issue
  - NTP Synchronization Lost → Time sync network issue

Impact: Service interruption, call setup failures
Investigation:
  1. Check network connectivity between devices
  2. Verify firewall rules allow required protocols
  3. Check peer device status and errors
  4. Review network interface statistics
```

Hardware & Environmental Alarms

```
Category: Physical Infrastructure
Probable Causes:
  - High Temperature Warning  → Cooling system degraded
  - Power Supply Degraded     → UPS/PSU issue
  - Fan Failure               → Cooling fan malfunction
  - Disk Space Low            → Storage approaching limit
  - Memory Exhaustion         → Process memory leak

Impact: Potential cascading failures, data loss
Investigation:
  1. Check hardware status via management interface
  2. Review temperature trends
  3. Check cooling system operation
  4. Monitor memory and CPU utilization
```

Software & Process Alarms

```
Category: Application Layer
Probable Causes:
  - Process Crash            → Software error or OOM
  - High CPU Utilization     → Performance bottleneck
  - Queue Overload           → Message processing backlog
  - License Violation        → Capacity exceeded

Impact: Service degradation or interruption
Investigation:
  1. Check process status
  2. Review logs for error messages
  3. Monitor CPU/memory/queue depth
  4. Verify license status
```

Radio Resource Alarms

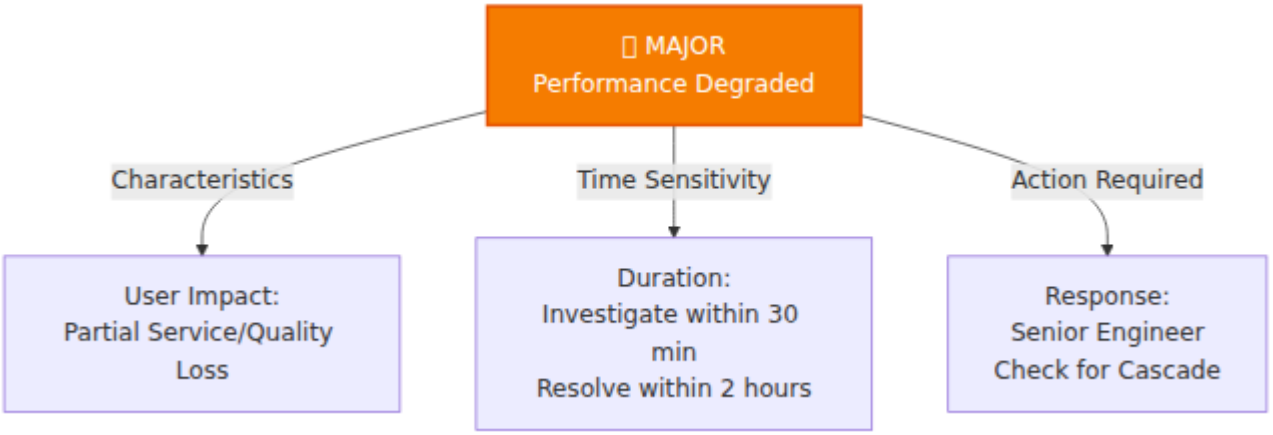
```
Category: Air Interface
Probable Causes:
  - Cell Unavailable         → No radio coverage/power
  - DL Resource Blocked      → Capacity exhaustion
  - Handover Failure Rate High → Neighbor configuration issue
  - Poor Cell Quality        → RF interference or path loss

Impact: User experience degradation
Investigation:
  1. Check cell physical parameters
  2. Review neighbor cell configuration
  3. Analyze RF quality metrics
  4. Check antenna alignment
```

Configuration Alarms

```
Category: System State
Probable Causes:
  - Parameter Mismatch       → Configuration inconsistency
  - License Expired          → Licensing issue
  - Config Checksum Error    → Corruption or conflict
  - Feature Not Licensed     → Feature usage exceeds license

Impact: Feature unavailability or degradation
Investigation:
  1. Review configuration changes
  2. Compare current vs. intended config
  3. Check license file and expiry
  4. Verify software version compatibility
```

Investigation & Troubleshooting

Investigation Workflow



Step 1: Review Alarm Details

When an alarm fires, start by gathering information:

What to collect:

- Alarm ID and unique identifier
- Severity and probable cause
- Affected system DN (device/cell/component)
- Event time (when it occurred)
- Duration (how long it's been active)
- Full alarm description and text

Tools:

- RAN Monitor Web UI → Alarms page
- Grafana → Active Alarms table
- InfluxDB → Query raw alarm record

Step 2: Research Probable Cause

Each alarm type has known causes and investigations:

Documented Knowledge:

- Nokia AirScale troubleshooting guides
- Previous ticket history (similar issues)
- RAN Monitor documented runbooks
- Team expertise (subject matter experts)

Step 3: Check Related Metrics

Cross-correlate alarms with performance metrics to understand impact:

```
Example: "DL Resource Blocked" Alarm
├─ Check DL Resource Utilization (should be > 95%)
├─ Check Traffic Throughput (trending up?)
├─ Check Call Setup Success (dropped?)
├─ Check Handover Success (affected?)
└─ Check Cell Availability (down?)
```

Tools:

- Grafana → Device-specific dashboard
- Web UI → Device detail page → Metrics section
- InfluxDB direct queries for correlation

Step 4: Correlate with Recent Changes

Many issues are caused by recent modifications:

```
Timeline:
├─ Configuration changes (last 4 hours)
├─ Software upgrades (last 24 hours)
├─ Feature parameter tuning (last 7 days)
├─ Maintenance activities (last 7 days)
├─ Network changes (last 7 days)
└─ Third-party changes (external network)
```

Tools:

- RAN Monitor → Configuration history
- Change management system
- Incident history (similar issues before)
- Cross-team notification logs

Step 5: Diagnose Root Cause

Based on investigation, identify the root cause:

Decision Tree Example: "Cell Unavailable" Alarm

```
Cell Unavailable Alarm
│
├─ Is device responding to management?
│  ├─ NO → Check device connectivity, reboot if needed
│  └─ YES → Continue
│
├─ Are all cells down or specific cell?
│  ├─ All cells → Check baseband/power supply hardware
│  └─ Specific cell → Continue
│
├─ Is cell transmitting power?
│  ├─ NO → Check Power Amplifier, antenna connection
│  └─ YES → Continue
│
├─ Are neighbor cells reporting this cell?
│  ├─ YES → Other devices see this cell as unavailable
│  │       → Check antenna alignment, cable connection
│  └─ NO → Cell is down for internal reason
│          → Check baseband module, DSP status
│
└─ Check logs for error messages
   → Software crash
   → License violation
   → Parameter error
```

Step 6: Implement Resolution

Once root cause is identified, implement fix:

Types of Resolutions:

Type	Method	Example
Immediate	Restart/reboot	Device restart to clear hung process
Configuration	Adjust parameters	Change handover threshold
Hardware	Replace/repair	Swap failed power supply
Software	Upgrade/patch	Install software bug fix
Network	Fix connectivity	Restore BGP route, fix firewall

Step 7: Verify Resolution

After implementing fix, verify:

```
Verification Checklist:
□ Alarm cleared in RAN Monitor
□ Related metrics have normalized
□ No secondary/cascading alarms
□ Performance metrics returned to baseline
□ Customer reports (if applicable) resolved
□ System is stable (> 30 minutes observation)
```

Step 8: Document Learning

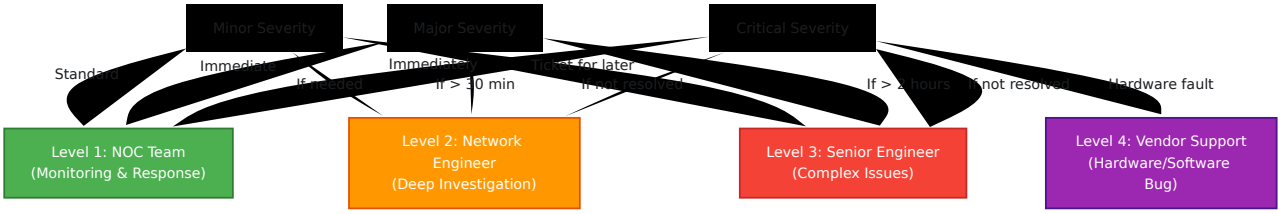
Record findings for future reference:

Document:

- Root cause and contributing factors
- Steps taken to resolve
- Time spent (for SLA tracking)
- Preventive measures taken
- Knowledge shared with team

Escalation Procedures

Escalation Ladder



Escalation Triggers

Escalate to Level 2 if:

- Critical alarm not cleared after 15 minutes
- Major alarm not cleared after 30 minutes
- Issue is outside NOC team's expertise
- Requires device reboot/major change
- Affects > 1 site simultaneously

Escalate to Level 3 if:

- Level 2 unable to diagnose after 1 hour
- Critical issue persists > 30 minutes
- Hardware failure suspected
- Cascading issues detected
- Requires vendor involvement

Contact Vendor (Level 4) if:

- Hardware failure confirmed (PSU, CMON, etc.)
- Software bug suspected (unrecoverable crash)
- License/activation issue
- Issue documented in vendor known issues
- Multiple escalation levels unable to resolve

Escalation Communication

Template for Escalating to Level 2:

```
Subject: Escalation - [Severity] - [Device] - [Issue]

Alert Time: [2025-12-10 14:30 UTC]
Duration: [15 minutes]
Device: [SITE_A_BS1]
Issue: [Cell Unavailable]

Symptoms:
- Cell A1 not responding to management
- All cells reporting cell unavailable
- Device ping successful

Investigation Performed:
- Device connectivity verified
- Baseband module status checked
- Power supply status: OK
- Temperature: Normal

Metrics:
- Cell availability: 0%
- No traffic on cell
- Control plane: Connected

Initial Analysis:
- Possible baseband module failure
- Or power amplifier hardware issue

Next Steps Recommended:
- Hardware diagnostics
- Module swap if available
- Device restart as last resort

Escalation Link: [Dashboard Link]
```

Resolution Tracking

SLA Tracking

Track time-to-resolution for SLA compliance:

```
Alarm Timeline:
├─ Event Time: 14:23:45 ← When issue occurred
├─ Detection Time: 14:23:47 (2 sec) ← Management detected
├─ Alert Time: 14:23:50 (5 sec) ← Operations notified
├─ Ack Time: 14:24:15 (30 sec) ← Operator acknowledged
├─ Investigation: 14:24:15 → 14:28:00 (3.75 min)
├─ Resolution: 14:28:00 → 14:28:35 (35 sec fix + verification)
└─ Clear Time: 14:28:36 ← Alarm cleared
    └─ Total Duration: 5 min 51 sec

SLA Metrics:
├─ Detection Latency: 2 seconds
├─ Alert-to-ACK: 30 seconds
├─ Time-to-Resolve: 5 min 51 sec
└─ SLA Status: PASS (< 15 min target)
```

Trend Analysis

Track patterns in alarm data:

Questions to ask:

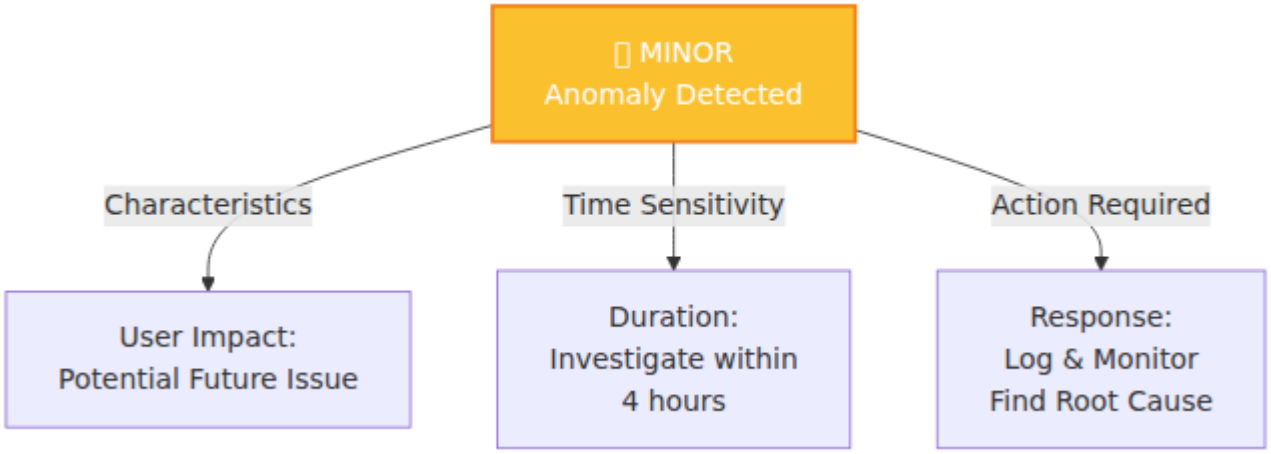
- Are we seeing the same alarm repeatedly?
- Is alarm rate increasing/decreasing?
- Do alarms cluster at specific times?
- Are multiple sites affected simultaneously?
- Is MTTR improving over time?

Tools:

- Grafana → Alarm trends dashboard
- Top alarms report (weekly)

- MTTR tracking by device/type

Preventing Repeat Issues



Best Practices

Operational Excellence

1. Alarm Fatigue Prevention

- Set meaningful thresholds (not too sensitive)
- Use duration windows (not single spike)
- Aggregate related alarms
- Suppress known false positives

2. Rapid Response

- Keep runbooks up-to-date
- Train team on common issues
- Use automation for routine resets
- Have escalation contacts readily available

3. Quality Documentation

- Document every incident

- Share learnings with team
- Update runbooks based on incidents
- Cross-train team members

4. Proactive Monitoring

- Watch for warnings before critical
- Trend analysis for capacity planning
- Regular health checks
- Performance baseline establishment

Runbook Development

Every frequent alarm should have a runbook:

Template:

```
Alarm: [Cell Unavailable]
Probability: [High]
MTTR: [5-15 minutes]
SLA Target: [Resolved within 30 minutes]

Symptoms:
- Alarm: "Cell Unavailable"
- Users: Unable to connect
- Metrics: Cell availability 0%

Quick Diagnosis (< 5 minutes):
1. Is device responding to ping?
2. Are other cells working?
3. Is baseband running (check logs)?

Resolution Steps:
Step 1: Device Connectivity Check
  - Ping device: ping 192.168.1.100
  - If no response → Check network connectivity

Step 2: Hardware Status
  - Check Power Amplifier status
  - Check baseband module LEDs
  - Check antenna connection

Step 3: Restart Cell
  - Restart cell via management interface
  - Wait 60 seconds for startup
  - Verify metrics normalize

Step 4: If Still Down
  - Escalate to Level 2
  - Prepare for device reboot
  - Notify customer

Escalation:
- If > 15 minutes → Escalate to [Engineer Name]
- If > 30 minutes → Escalate to [Senior Engineer]
- If hardware failure → Contact Nokia Support

Prevention:
- Regular baseband firmware updates
```

```
- Preventive power supply replacement
- Antenna connection inspection quarterly
```

API Endpoints & Configuration Management

REST API for Managing RAN Device Configuration and Operations

Guide to managing base station configuration, querying device state, and automating RAN operations using the vendor-agnostic API

Table of Contents

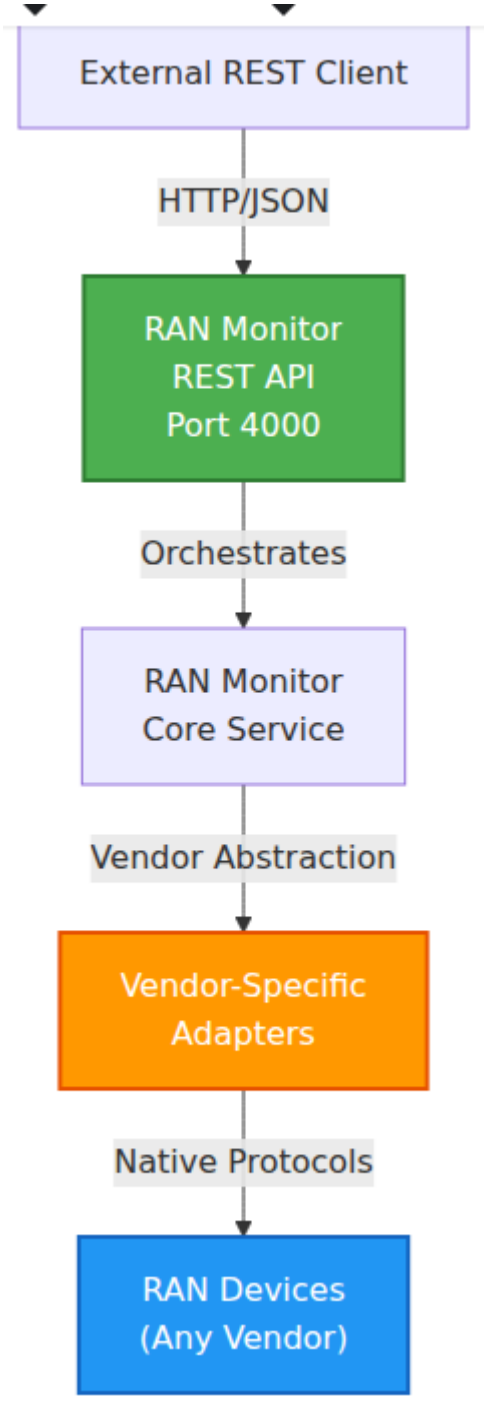
Overview

RAN Monitor exposes a comprehensive REST API for managing RAN device configuration and querying operational data. The API provides a vendor-agnostic interface that abstracts the underlying device communication protocols. The API enables:

- **Device Management** - Register, unregister, and monitor devices

- **Configuration Queries** - Retrieve device parameters and system state
- **Data Collection** - Pull performance metrics, alarms, and topology
- **Session Control** - Manage communication sessions with devices
- **Network Operations** - Automate routine management tasks

API Architecture



API Features

- **RESTful Design** - Standard HTTP methods (GET, POST, PUT, DELETE)
- **JSON Format** - All requests and responses are JSON
- **Vendor Abstraction** - Unified API across different device vendors
- **Stateful Operations** - Maintains device sessions and state
- **Error Handling** - Detailed error messages and status codes
- **Async Processing** - Non-blocking requests for long operations

---

Authentication & Access

Device Registration

Before any operations, a device must be registered with RAN Monitor. Registration establishes the connection between RAN Monitor and the device using each vendor's native authentication mechanism.

Registration Process:

1. Device credentials (username/password or API keys) are securely stored
2. Initial connectivity test verifies device is reachable
3. Device is registered and ready for operations
4. Continuous health monitoring begins

API Access Control

Currently, RAN Monitor API is accessible within the management network. For production deployments, consider:

- **Authentication Methods:**
  - API key in header: `Authorization: Bearer <api-key>`
  - OAuth2 for integration with identity providers
  - Network-based access control (firewall/VPN)

- **Rate Limiting:**
  - Per-client limits to prevent abuse
  - Per-device limits for operation frequency
- **Audit Logging:**
  - All API calls are logged with timestamps and user/client info
  - Configuration changes are tracked with before/after values

---

API Architecture

REST API Endpoints

The RAN Monitor API is organized into resource-based endpoints:

```
/api/v1/
├── /devices                    # Device management
│   ├── GET                     # List all devices
│   ├── POST                    # Register new device
│   └── /:id                    # Device operations
│       ├── GET                 # Get device details
│       ├── PUT                 # Update device config
│       ├── DELETE              # Remove device
│       ├── /sessions           # Session management
│       ├── /config             # Configuration operations
│       ├── /alarms             # Alarm queries
│       └── /metrics            # Performance data
│
├── /operations                 # Bulk operations
│   ├── /get                    # Retrieve data
│   ├── /set                    # Modify configuration
│   └── /search                 # Search devices
│
└── /health                     # System health
    ├── /status                 # Overall status
    └── /devices/:id/ping       # Device connectivity
```

Base URL

```
http://<ran-monitor-ip>:4000/api/v1
```

Endpoint Reference

Device Management

List All Devices

```
GET /api/v1/devices
```

Response:


```
{
  "devices": [
    {
      "id": "nokia_bs1",
      "name": "SITE_A_BS1",
      "vendor": "Nokia",
      "address": "192.168.1.100",
      "port": 8080,
      "status": "registered",
      "registered_at": "2025-12-10T14:30:00Z",
      "session_active": true,
      "software_version": "BSC-2250.5.0",
      "license_required": false
    }
  ]
}
```


Get Device Details

```
GET /api/v1/devices/:id
```

Response:


```
{
  "device": {
    "id": "nokia_bs1",
    "name": "SITE_A_BS1",
    "vendor": "Nokia",
    "address": "192.168.1.100",
    "registration_status": "registered",
    "registration_key": "base64_encoded_key",
    "session_id": "nonuniquesession",
    "session_expiry": "2025-12-11T14:30:00Z",
    "device_info": {
      "type": "AirScale",
      "software_release": "5.0.0",
      "hardware_version": "2.0",
      "agent_unique_id": "airscale-001"
    }
  }
}
```


Register a Device

```
PUT /api/v1/devices/:id/register
Content-Type: application/json

{
  "address": "192.168.1.100:8080",
  "web_username": "admin",
  "web_password": "password",
  "webhook_url": "http://manager.example.com:9076/webhook",
  "private_key_path": "/etc/certs/private.key",
  "public_key_path": "/etc/certs/public.key"
}
```

Response:

```
{
  "result": "Success",
  "registration_key": "base64_encoded_nonce",
  "device_id": "nokia_bs1",
  "message": "Device registered successfully"
}
```

Status Codes:

- `200` - Registration successful
- `400` - Invalid parameters or device error
- `409` - Device already registered
- `500` - Internal error

Unregister a Device

```
DELETE /api/v1/devices/:id
```

Response:

```
{
  "result": "Success",
  "message": "Device unregistered",
  "device_id": "nokia_bs1"
}
```

Session Management

Start a Session

```
PUT /api/v1/devices/:id/sessions
Content-Type: application/json

{
  "session_id": "session_unique_identifier"
}
```

Response:

```
{
  "result": "Success",
  "session_id": "session_unique_identifier",
  "session_timeout": 86400,
  "expires_at": "2025-12-11T14:30:00Z"
}
```

Session Lifetime:

- Default timeout: 24 hours
- Keep-alive required before timeout
- Automatic refresh every 20 hours

Check Session Status

```
GET /api/v1/devices/:id/sessions
```

Response:

```
{
  "session": {
    "active": true,
    "session_id": "session_unique_identifier",
    "expires_at": "2025-12-11T14:30:00Z",
    "time_remaining_seconds": 82400,
    "last_activity": "2025-12-10T14:30:00Z"
  }
}
```

Keep-Alive Session

```
POST /api/v1/devices/:id/sessions/keep-alive
```

Response:

```
{
  "result": "Success",
  "new_expiry": "2025-12-11T14:30:00Z"
}
```

Configuration Management

Query Configuration

Retrieve device configuration parameters:

```
PUT /api/v1/devices/:id/config/upload
Content-Type: application/json

{
  "filter": {
    "uploadType": "configuration",
    "objects": [
      {
        "sdn": "/BSC-1/BTS-23/*",
        "depth": 100
      }
    ],
    "objectClass": ""
  }
}
```

Response:

```
{
  "result": "Success",
  "configuration": {
    "timestamp": "2025-12-10T14:30:00Z",
    "device_id": "nokia_bs1",
    "parameters": {
      "/BSC-1/BTS-23": {
        "BtsBasics": {
          "BtsName": "CELL_A",
          "BtsType": "MACRO",
          "EnvironmentalSpecifications": {
            "TemperatureRange": "Industrial"
          }
        },
        "CarrierAggregation": {
          "CarrierAggregationCapability": true,
          "MaxUECarriers": 5
        }
      }
    }
  }
}
```


Set Configuration Parameter

```
PUT /api/v1/devices/:id/config/set
Content-Type: application/json

{
  "parameter_path": "/BSC-1/BTS-23/BtsBasics/BtsName",
  "value": "NEW_CELL_NAME",
  "value_type": "string"
}
```

Response:

```
{
  "result": "Success",
  "parameter": "/BSC-1/BTS-23/BtsBasics/BtsName",
  "old_value": "CELL_A",
  "new_value": "NEW_CELL_NAME",
  "applied_at": "2025-12-10T14:30:45Z"
}
```


Common Configuration Parameters:

Parameter	Type	Example	Purpose
BtsName	String	"SITE_A_Cell_1"	Cell/base station identifier
MaxUEsServed	Integer	256	Maximum simultaneous UEs
CellTXPower	Integer	40 (dBm)	Transmit power level
EnableCarrierAgg	Boolean	true	Carrier aggregation support
HandoverHysteresis	Integer	3 (dB)	Handover sensitivity

Get Configuration History

```
GET /api/v1/devices/:id/config/history?limit=10&days=7
```

Response:

```
{
  "history": [
    {
      "timestamp": "2025-12-10T14:30:45Z",
      "change_type": "parameter_modified",
      "parameter": "/BSC-1/BTS-23/BtsBasics/BtsName",
      "old_value": "CELL_A",
      "new_value": "NEW_CELL_NAME",
      "reason": "Manual config update"
    }
  ]
}
```

Data Retrieval

Get Performance Metrics

Retrieve performance counter data:

```
PUT /api/v1/devices/:id/metrics/upload
Content-Type: application/json

{
  "filter": {
    "uploadType": "measurement",
    "objects": [
      {
        "sdn": "*",
        "depth": 100
      }
    ]
  }
}
```

Response:

```
{
  "result": "Success",
  "metrics": {
    "timestamp": "2025-12-10T14:30:00Z",
    "measurement_interval": 300,
    "counters": [
      {
        "id": "M1C1",
        "name": "DL Cell Throughput",
        "value": 125.4,
        "unit": "Mbps",
        "cell_dn": "/BSC-1/BTS-23/Cell-1"
      },
      {
        "id": "M1C2",
        "name": "UL Cell Throughput",
        "value": 89.2,
        "unit": "Mbps",
        "cell_dn": "/BSC-1/BTS-23/Cell-1"
      }
    ]
  }
}
```


Get Active Alarms

```
PUT /api/v1/devices/:id/alarms/upload
Content-Type: application/json

{
  "filter": {
    "uploadType": "active_faults"
  }
}
```

Response:

```
{
  "result": "Success",
  "alarms": [
    {
      "alarm_id": "a1b2c3d4",
      "severity": "Critical",
      "probable_cause": "Cell Unavailable",
      "specific_problem": "Power Supply Failure",
      "affected_dn": "/BSC-1/BTS-23/Cell-1",
      "event_time": "2025-12-10T14:15:30Z",
      "description": "Cell 1 is unavailable due to power supply
failure"
    }
  ]
}
```


Get Device Topology


```
PUT /api/v1/devices/:id/topology/upload
Content-Type: application/json

{
  "filter": {
    "uploadType": "topology",
    "objects": [
      {
        "sdn": "*",
        "depth": 100
      }
    ]
  }
}
```


Response:

```
{
  "result": "Success",
  "topology": {
    "device_dn": "/BSC-1",
    "managed_elements": [
      {
        "name": "BTS-23",
        "type": "BaseTransceiverStation",
        "dn": "/BSC-1/BTS-23",
        "cells": [
          {
            "name": "Cell-1",
            "type": "EUtranCell",
            "physical_cell_id": 100,
            "frequency": 2110
          }
        ]
      }
    ]
  }
}
```

Health Checks

Ping Device

```
PUT /api/v1/devices/:id/ping
```

Response:


```
{
  "result": "Success",
  "device_id": "nokia_bs1",
  "latency_ms": 45,
  "status": "reachable"
}
```


Get System Health

```
GET /api/v1/health/status
```

Response:

```
{
  "status": "healthy",
  "devices": {
    "total": 50,
    "registered": 48,
    "active_sessions": 45,
    "unreachable": 2
  },
  "database": {
    "mysql": "connected",
    "influxdb": "connected"
  },
  "timestamp": "2025-12-10T14:30:00Z"
}
```

Configuration Management

Configuration Data Model

Nokia eNodeB configuration is organized hierarchically:

```
/SystemFunctions
├── /BSC-1 (Base Station Controller)
│   └── /BTS-23 (Base Transceiver Station)
│       ├── BtsBasics (Name, type, location)
│       ├── /Cell-1
│       │   ├── CellCommonData
│       │   ├── CellAdvanced
│       │   └── CarrierAggregation
│       └── /Cell-2
│           └── ...
└── /Connectivity
    ├── S1Interface
    ├── X2Interface
    └── NetworkConfiguration
```

Common Configuration Tasks

Enable/Disable a Cell

```
{
  "parameter_path": "/BSC-1/BTS-23/Cell-
1/CellCommonData/AdminState",
  "value": "UNLOCKED",
  "value_type": "enum"
}
```

Possible values: `LOCKED`, `UNLOCKED`, `SHUTTING_DOWN`

Adjust Cell Power

```
{
  "parameter_path": "/BSC-1/BTS-23/Cell-
1/CellAdvanced/CellTXPower",
  "value": "35",
  "value_type": "integer"
}
```

Range: 0-46 dBm (device-dependent)

Configure Handover Hysteresis

```
{
  "parameter_path": "/BSC-1/BTS-23/Cell-
1/CellAdvanced/HandoverHysteresis",
  "value": "3",
  "value_type": "integer"
}
```

Unit: dB, typical range: 0-8 dB

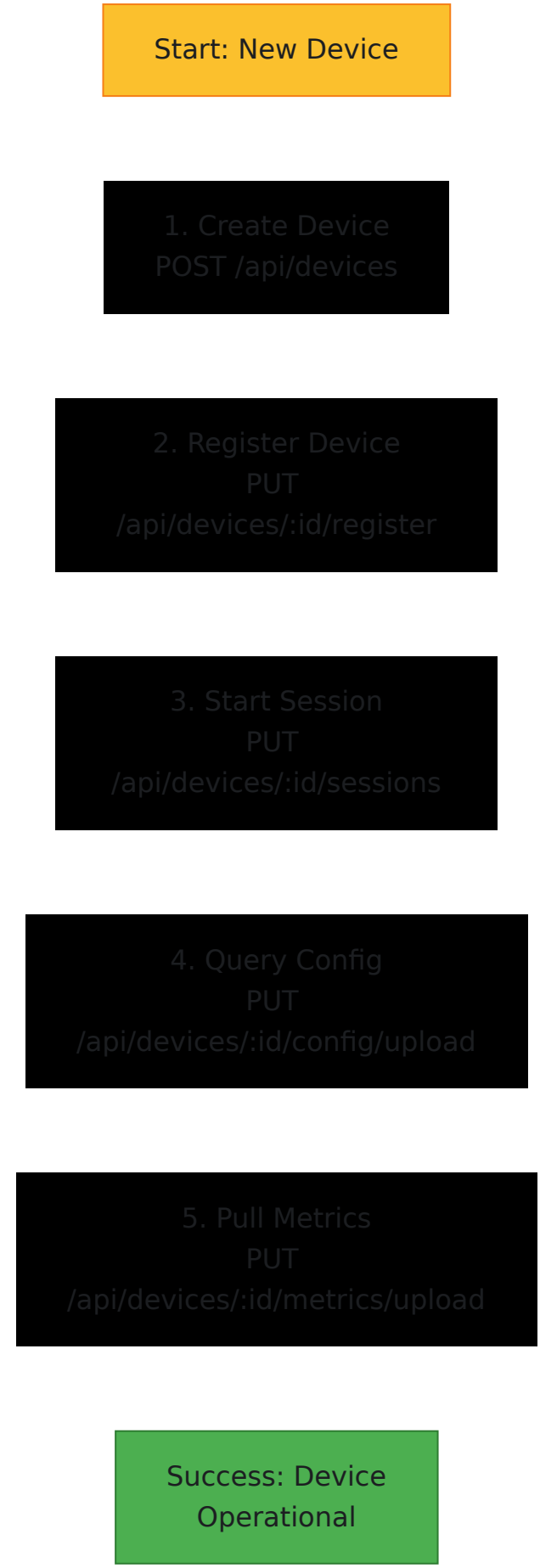
Set Maximum Connected Users

```
{
  "parameter_path": "/BSC-1/BTS-23/MaxUEsServed",
  "value": "256",
  "value_type": "integer"
}
```

Device-dependent limit

Common Workflows

Workflow 1: Device Onboarding


Start: New Device
1. Create Device
POST /api/devices
2. Register Device
PUT
/api/devices/:id/register
3. Start Session
PUT
/api/devices/:id/sessions
4. Query Config
PUT
/api/devices/:id/config/upload
5. Pull Metrics
PUT
/api/devices/:id/metrics/upload
Success: Device
Operational

Example:

```
# 1. Create device entry
curl -X POST http://localhost:4000/api/v1/devices \
  -H "Content-Type: application/json" \
  -d '{
    "id": "site_a_bs1",
    "name": "SITE_A_BS1",
    "vendor": "Nokia",
    "address": "192.168.1.100:8080",
    "credentials": {
      "username": "admin",
      "password": "password"
    }
  }'

# 2. Register with device
curl -X PUT
http://localhost:4000/api/v1/devices/site_a_bs1/register \
  -H "Content-Type: application/json" \
  -d '{
    "webhook_url": "http://manager.example.com:9076/webhook"
  }'

# 3. Start session
curl -X PUT
http://localhost:4000/api/v1/devices/site_a_bs1/sessions \
  -H "Content-Type: application/json" \
  -d '{"session_id": "session_001"}'

# 4. Get configuration
curl -X PUT
http://localhost:4000/api/v1/devices/site_a_bs1/config/upload \
  -H "Content-Type: application/json" \
  -d '{
    "filter": {
      "uploadType": "configuration",
      "objects": [{"sdn": "*", "depth": 100}]
    }
  }'
```

Workflow 2: Configuration Update

```
# 1. Query current value
curl -X PUT
http://localhost:4000/api/v1/devices/site_a_bs1/config/upload \
  -H "Content-Type: application/json" \
  -d '{
    "filter": {
      "uploadType": "configuration",
      "objects": [{"sdn": "/BSC-1/BTS-23/Cell-1", "depth": 10}]
    }
  }' | jq '.configuration.parameters["/BSC-1/BTS-23/Cell-1"]'

# 2. Modify parameter
curl -X PUT
http://localhost:4000/api/v1/devices/site_a_bs1/config/set \
  -H "Content-Type: application/json" \
  -d '{
    "parameter_path": "/BSC-1/BTS-23/Cell-
1/CellAdvanced/CellTXPower",
    "value": "38",
    "value_type": "integer"
  }'

# 3. Verify change
curl -X PUT
http://localhost:4000/api/v1/devices/site_a_bs1/config/upload \
  -H "Content-Type: application/json" \
  -d '{
    "filter": {
      "uploadType": "configuration",
      "objects": [{"sdn": "/BSC-1/BTS-23/Cell-1/CellAdvanced",
"depth": 5}]
    }
  }' | jq '.configuration.parameters["/BSC-1/BTS-23/Cell-
1/CellAdvanced/CellTXPower"]'
```

Workflow 3: Performance Monitoring

```
# Continuous monitoring loop (example script)
#!/bin/bash

DEVICE="site_a_bs1"
INTERVAL=300  # 5 minutes

while true; do
  # Pull metrics
  METRICS=$(curl -s -X PUT
http://localhost:4000/api/v1/devices/$DEVICE/metrics/upload \
    -H "Content-Type: application/json" \
    -d '{
      "filter": {
        "uploadType": "measurement",
        "objects": [{"sdn": "*", "depth": 100}]
      }
    }')

  # Extract key metrics
  DL=$(echo $METRICS | jq '.metrics.counters[] |
select(.id=="M1C1") | .value')
  CELLS=$(echo $METRICS | jq '.metrics.counters | length')

  echo "$(date): DL=$DL Mbps, Cells=$CELLS"

  # Check alarms
  ALARMS=$(curl -s -X PUT
http://localhost:4000/api/v1/devices/$DEVICE/alarms/upload \
    -H "Content-Type: application/json" \
    -d '{
      "filter": {
        "uploadType": "active_faults"
      }
    }' | jq '.alarms | length')

  if [ "$ALARMS" -gt 0 ]; then
    echo "WARNING: $ALARMS active alarms"
  fi
```

```
  sleep $INTERVAL
done
```

Error Handling

HTTP Status Codes

Code	Meaning	Example
`200`	Success	Configuration retrieved
`201`	Created	Device registered
`400`	Bad Request	Invalid JSON or parameters
`401`	Unauthorized	Missing/invalid API key
`404`	Not Found	Device doesn't exist
`409`	Conflict	Device already registered
`500`	Server Error	Database connection failure
`503`	Unavailable	Maintenance mode

Error Response Format

```
{
  "error": {
    "code": "DEVICE_NOT_FOUND",
    "message": "Device 'site_a_bs1' not found",
    "details": {
      "device_id": "site_a_bs1",
      "timestamp": "2025-12-10T14:30:00Z"
    }
  }
}
```

Common Errors

Device Not Registered:

```
{
  "error": {
    "code": "NOT_REGISTERED",
    "message": "Device must be registered before operations",
    "solution": "Call PUT /api/devices/:id/register first"
  }
}
```

Session Expired:

```
{
  "error": {
    "code": "SESSION_EXPIRED",
    "message": "Device session has expired",
    "solution": "Call PUT /api/devices/:id/sessions to start new
session"
  }
}
```

Configuration Parameter Invalid:

```
{
  "error": {
    "code": "INVALID_PARAMETER",
    "message": "Parameter value out of range",
    "details": {
      "parameter": "/BSC-1/BTS-23/Cell-
1/CellAdvanced/CellTXPower",
      "value": "99",
      "valid_range": "0-46 dBm"
    }
  }
}
```

API Examples

Python Client Example

```
import requests
import json

class RanMonitorClient:
    def __init__(self, base_url="http://localhost:4000/api/v1"):
        self.base_url = base_url
        self.session = requests.Session()

    def register_device(self, device_id, address, username,
password):
        """Register a new device"""
        url = f"{self.base_url}/devices/{device_id}/register"
        payload = {
            "address": address,
            "web_username": username,
            "web_password": password,
            "webhook_url": "http://manager:9076/webhook"
        }
        response = self.session.put(url, json=payload)
        return response.json()

    def get_config(self, device_id, sdn="*", depth=100):
        """Retrieve device configuration"""
        url = f"{self.base_url}/devices/{device_id}/config/upload"
        payload = {
            "filter": {
                "uploadType": "configuration",
                "objects": [{"sdn": sdn, "depth": depth}]
            }
        }
        response = self.session.put(url, json=payload)
        return response.json()

    def set_config(self, device_id, parameter_path, value,
value_type="string"):
        """Update a configuration parameter"""
        url = f"{self.base_url}/devices/{device_id}/config/set"
        payload = {
```

```
            "parameter_path": parameter_path,
            "value": value,
            "value_type": value_type
        }
        response = self.session.put(url, json=payload)
        return response.json()

    def get_metrics(self, device_id):
        """Retrieve performance metrics"""
        url = f"
{self.base_url}/devices/{device_id}/metrics/upload"
        payload = {
            "filter": {
                "uploadType": "measurement",
                "objects": [{"sdn": "*", "depth": 100}]
            }
        }
        response = self.session.put(url, json=payload)
        return response.json()

# Example usage
client = RanMonitorClient()

# Register device
result = client.register_device(
    device_id="site_a_bs1",
    address="192.168.1.100:8080",
    username="admin",
    password="password"
)
print(f"Registration: {result}")

# Get configuration
config = client.get_config("site_a_bs1")
print(f"Config: {json.dumps(config, indent=2)}")

# Update parameter
update = client.set_config(
    "site_a_bs1",
    "/BSC-1/BTS-23/Cell-1/CellAdvanced/CellTXPower",
    "38",
    "integer"
```

```
)
print(f"Update: {update}")
```

cURL Examples

Register Device:

```
curl -X PUT
http://localhost:4000/api/v1/devices/site_a_bs1/register \
  -H "Content-Type: application/json" \
  -d '{
    "address": "192.168.1.100:8080",
    "web_username": "admin",
    "web_password": "password",
    "webhook_url": "http://manager:9076/webhook"
  }'
```

Get Device Status:

```
curl -X GET http://localhost:4000/api/v1/devices/site_a_bs1
```

Query Configuration:

```
curl -X PUT
http://localhost:4000/api/v1/devices/site_a_bs1/config/upload \
  -H "Content-Type: application/json" \
  -d '{
    "filter": {
      "uploadType": "configuration",
      "objects": [{"sdn": "/BSC-1/*", "depth": 50}]
    }
  }' | jq '.'
```

Configuration Archive Guide

Automated Versioning and Historical Tracking for AirScale Configurations

---

Overview

The Configuration Archive system automatically tracks and versions all AirScale base station configuration files. Instead of storing configuration snapshots in InfluxDB, configurations are saved as timestamped XML files on the server, providing a complete audit trail of configuration changes.

Key Features

- **Automatic Versioning** - New versions created only when configuration changes
- **Hourly Polling** - Checks for config changes every hour (configurable)
- **Change Detection** - Smart comparison detects actual changes, ignoring whitespace
- **Size-Based Limits** - Maximum 100 MB storage per device (keeps ~690 versions)
- **Web Interface** - Browse, download, and manage config versions
- **Fast Access** - File-based storage for instant retrieval
- **Zero InfluxDB Load** - Configs no longer stored in time-series database
- **Automatic Cleanup** - Old versions deleted when size limit reached

---

How It Works

Polling Schedule

Configuration is polled from each registered AirScale base station:

- **Interval:** Every 1 hour (default)
- **First Poll:** Immediately on application startup
- **Change Detection:** Compares content with previous version
- **Storage:** Only saves if changed or first time

Storage Location

Configurations are stored on the RAN Monitor server filesystem:

```
priv/airscale_configs/
└── <airscale-name>/
    ├── current.xml                                    # Latest
configuration
    ├── ONS-Lab-Airscale_config_20251230_143522.xml   # Version
from Dec 30, 2:35pm
    ├── ONS-Lab-Airscale_config_20251229_120000.xml   # Previous
version
    ├── ONS-Lab-Airscale_config_20251228_093045.xml   # Older
version
    └── ...                                            # Versions
kept until 100 MB limit
```

Filename Format: `<AirScaleName>_config_YYYYMMDD_HHMMSS.xml`

Directory Naming: AirScale names are sanitized (special characters replaced with underscores, lowercase)

Version Management

- **Latest Version:** Always available as `current.xml`
- **Historical Versions:** Timestamped files showing when config changed

- **Auto-Cleanup:** Deletes oldest versions when 100 MB size limit reached
- **Manual Cleanup:** Delete specific versions via Web UI (except `current.xml`)
- **Storage Protection:** Size-based limit prevents unlimited disk usage
- **Flexible Retention:** More versions if files are small, fewer if files are large

---

Storage Protection

Size-Based Storage Limit

To prevent unlimited disk usage, the system uses a **size-based limit** instead of a version count:

- **Maximum Size:** 100 MB per device (configurable)
- **Automatic Cleanup:** Oldest versions deleted when size limit exceeded
- **Cleanup Timing:** Every time a new config version is saved
- **Protected Files:** `current.xml` and at least one version always kept
- **Flexible:** Keeps ~690 versions at 145KB each, more if files are smaller

How It Works

When a new config version is saved:

1. **Save new version** - Config written as `<AirScale>_config_YYYYMMDD_HHMMSS.xml`
2. **Update current** - `current.xml` updated with latest config
3. **Calculate size** - System sums total size of all versioned files
4. **Cleanup old** - If total > 100 MB, delete oldest versions until under limit
5. **Log activity** - Deletions logged with freed space

Example Scenario

```
Initial state: 95 MB used (655 versions at 145KB each)
├── ONS-Lab-Airscale_config_20240101_100000.xml  <- Oldest (145KB)
├── ONS-Lab-Airscale_config_20240102_100000.xml  (145KB)
├── ... (653 more versions)
└── ONS-Lab-Airscale_config_20251230_100000.xml  <- Newest (145KB)

New config detected at 2025-12-31 10:00:00 (145KB)

Actions:
1. Save: ONS-Lab-Airscale_config_20251231_100000.xml (145KB)
2. Total size now: 95 MB + 145KB = 95.14 MB (still under 100 MB
limit)
3. No deletion needed
4. Final: 656 versions, 95.14 MB used

---

Later: Large config change (new features added, file is now 500KB)

Actions:
1. Save: ONS-Lab-Airscale_config_20251231_150000.xml (500KB)
2. Total size now: 95.14 MB + 500KB = 95.64 MB (still under limit)
3. No deletion needed
4. Final: 657 versions, 95.64 MB used

---

After many more changes: Approaching limit

Current state: 99.8 MB (685 versions)

New config: 200KB
1. Save new version
2. Total would be: 100 MB (exceeds limit)
3. Delete oldest versions until total < 100 MB
4. Deletions logged: "Deleted 3 versions, freed 435 KB"
5. Final: 682 versions, 99.6 MB used
```

Storage Guarantees

The automatic cleanup ensures:

- **Bounded Storage:** Each device limited to 100 MB maximum
- **No Surprises:** Storage won't grow indefinitely
- **Production Safe:** No manual intervention required
- **Flexible History:** More versions for small configs, fewer for large configs
- **Always Available:** At least one version always kept

Monitoring Storage

Check storage usage for all devices:

```
# Total storage used
du -sh priv/airscale_configs/
# Example: 215M (for 3 devices averaging 70 MB each)

# Storage per device
du -sh priv/airscale_configs/*/
# Example:
# 95M     priv/airscale_configs/ons-lab-airscale/
# 68M     priv/airscale_configs/sector-1/
# 52M     priv/airscale_configs/sector-2/

# Check if any device is near the limit
find priv/airscale_configs -maxdepth 1 -type d -exec du -sm {} \;
| awk '$1 > 90 {print $2 " is at " $1 "MB (approaching 100MB
limit)"}'
```

Using the Config Archive Page

Accessing Config Archive

Web UI: Navigate to **Nokia → Config Archive**

URL: `https://<ran-monitor-ip>:9443/nokia/config-archive`

The Config Archive interface showing base station selector, version history table with timestamps and file sizes, and storage information.

Interface Overview

The Config Archive page has three main sections:

1. Base Station Selector

- **Grid View** - Shows all registered AirScale devices
- **Version Count** - Number of config versions stored for each
- **Selection** - Click a device to view its config history
- **Visual Indicator** - Selected device highlighted in blue

2. Version History Table

Displays all configuration versions for the selected base station:

Column	Description
Timestamp	When the configuration was saved (UTC)
Filename	Version filename (e.g., `config_20251230_143522.xml`)
Size	File size in KB or MB
Age	How long ago the version was created (e.g., "2h ago", "3d ago")
Actions	Download or Delete buttons

Sorting: Newest versions appear first (descending by timestamp)

Current Config: The `current.xml` file cannot be deleted (safety measure)

3. Storage Information

Summary panel showing:

- **Total Versions** - Number of config versions stored
- **Total Size** - Combined size of all versions
- **Storage Path** - Filesystem location on server

Common Operations

Downloading a Configuration

Purpose: Retrieve a specific config version for review, backup, or comparison

Steps:

1. Navigate to Config Archive page
2. Select the desired base station

3. Find the version you want in the table
4. Click **Download** button
5. File downloads with format: `<AirScaleName>_config_YYYYMMDD_HHMMSS.xml` (matches stored filename)

Use Cases:

- Creating offline backups
- Comparing configurations between timestamps
- Rolling back to previous known-good config
- Analyzing configuration drift over time

Deleting Old Versions

Purpose: Remove outdated config versions to free storage space

Steps:

1. Navigate to Config Archive page
2. Select the base station
3. Find the version to delete
4. Click **Delete** button
5. Confirm deletion in popup dialog
6. Version is permanently removed

Important Notes:

- Cannot delete `current.xml` (latest version protected)
- Deletion is immediate and cannot be undone
- Manual deletion doesn't affect auto-cleanup settings

Comparing Configurations

Purpose: Identify what changed between two config versions

Manual Comparison:

1. Download both versions you want to compare
2. Use an XML diff tool (e.g., `xmldiff`, `Beyond Compare`, `WinMerge`)
3. Review differences to understand changes

Example using command line:

```
# Download both versions
wget https://<server>:9443/download/config/ONS-Lab-Airscale/ONS-
Lab-Airscale_config_20251230_143522.xml
wget https://<server>:9443/download/config/ONS-Lab-Airscale/ONS-
Lab-Airscale_config_20251229_120000.xml

# Compare with diff
diff ONS-Lab-Airscale_config_20251229_120000.xml ONS-Lab-
Airscale_config_20251230_143522.xml

# Or use xmldiff for cleaner output
xmldiff ONS-Lab-Airscale_config_20251229_120000.xml ONS-Lab-
Airscale_config_20251230_143522.xml
```

Configuration Management Workflows

Audit Trail Investigation

Scenario: Need to determine when a configuration changed

Steps:

1. Open Config Archive
2. Select affected base station
3. Review version timestamps
4. Download relevant versions
5. Compare to identify exact changes
6. Correlate with performance issues or alarms

Example:

```
Version Timeline:
- ONS-Lab-Airscale_config_20251230_143522.xml (143KB) - Most
recent
- ONS-Lab-Airscale_config_20251228_091045.xml (142KB) - 2 days ago
- ONS-Lab-Airscale_config_20251225_180000.xml (142KB) - 5 days ago

Analysis:
- Size increased from 142KB to 143KB on Dec 30
- Compare Dec 28 vs Dec 30 to find what was added
- Check if timing correlates with alarm spike
```

Configuration Rollback

Scenario: Recent config change caused issues, need to restore previous version

Steps:

1. Identify known-good configuration version
2. Download that version from Config Archive
3. Navigate to Configuration Management page (Web UI)
4. Upload the downloaded config → receive Plan ID
5. Validate the plan → check for errors
6. If validation succeeds, activate the plan
7. Monitor device for stability
8. Verify new config appears in archive after next poll

Safety Checklist:

- ✓ Downloaded correct previous version
- ✓ Validated plan before activation
- ✓ Coordinated with operations team
- ✓ Scheduled during maintenance window
- ✓ Monitoring tools ready for verification

Baseline Configuration Management

Scenario: Maintain a "golden config" for standardization

Best Practice:

1. Create and validate baseline configuration
2. Apply to reference device
3. Download from Config Archive after next poll
4. Store externally as template
5. Use for deploying new devices
6. Periodically review and update baseline

Technical Details

Change Detection Algorithm

The system uses smart content comparison to avoid false positives:

Normalization Process:

1. Remove leading/trailing whitespace
2. Collapse whitespace between XML tags
3. Normalize internal whitespace
4. Compare resulting normalized content

Benefits:

- Formatting changes don't trigger new versions
- Only actual configuration changes create versions
- Reduces storage requirements
- Provides meaningful change history

Example:

```
<!-- These are considered identical -->

<!-- Version 1 (with extra whitespace) -->
<parameter>
    <name>cellId</name>
    <value>1</value>
</parameter>

<!-- Version 2 (compact) -->
<parameter><name>cellId</name><value>1</value></parameter>
```

Storage Requirements

Typical Config Size: ~145 KB per version (based on actual AirScale configs)

Capacity Planning:

Devices	Max Size per Device	Versions Kept (at 145KB)	Max Total Storage
10	100 MB	~690	1 GB
50	100 MB	~690	5 GB
100	100 MB	~690	10 GB
500	100 MB	~690	50 GB
1000	100 MB	~690	100 GB

Growth Characteristics:

- **Maximum storage per device:** 100 MB (configurable)
- **Typical versions retained:** ~690 (145KB each)
- **If config never changes:** Minimal growth (only current.xml at 145KB)
- **If config changes frequently:** Growth stops at 100 MB limit
- **Adaptive:** Keeps more versions for small configs, fewer for large configs

Automatic Protection:

- Old versions deleted when size limit reached
- No manual intervention needed
- Storage usage strictly capped per device
- At least one version always retained

Retention Policy

Default Settings:

- Maximum **100 MB** storage per device
- Automatically delete oldest versions when limit exceeded
- `current.xml` always retained (exempt from cleanup)
- Cleanup occurs every time a new version is saved
- At least one versioned file always kept

Customizing Retention:

The storage limit is configured in the ConfigStorage module:

```
# In lib/ran_monitor/nokia/airscale/config_storage.ex
# Module attribute at the top of the file
@max_storage_bytes 100 * 1024 * 1024  # 100 MB default

# Change to different limit:
@max_storage_bytes 50 * 1024 * 1024   # 50 MB (keeps ~345
versions)
@max_storage_bytes 200 * 1024 * 1024  # 200 MB (keeps ~1380
versions)

# The cleanup function uses this default
def cleanup_old_versions(airscale_name, max_size_bytes \
@max_storage_bytes)
```

After modifying, recompile:

```
mix compile
# Restart RAN Monitor to apply changes
```

Polling Configuration

Default Interval: 1 hour (3,600,000 milliseconds)

To Change Polling Interval:

Edit `lib/ran_monitor/nokia/airscale/manager.ex`:

```
defp schedule_get_airscale_config do
  # Pull config every 1 hour (3,600,000 ms)
  Process.send_after(self(), :get_airscale_config, 3_600_000)
end
```

Common Intervals:

- 30 minutes: `1_800_000`
- 1 hour: `3_600_000` (default)
- 2 hours: `7_200_000`
- 4 hours: `14_400_000`
- 24 hours: `86_400_000`

After changing, recompile and restart RAN Monitor.

Troubleshooting

No Config Versions Showing

Symptoms:

- Config Archive page shows "0 versions"
- Selected device shows empty table

Possible Causes:

1. **Device Not Registered**
   - Check Base Stations page
   - Verify device shows "REGISTERED" status
   - Review Application Logs for registration errors
2. **Session Not Active**
   - Check device detail view
   - Ensure session status is "ACTIVE"
   - Review session timestamps
3. **Config Not Yet Polled**
   - Wait for first hourly poll cycle
   - Or manually trigger: `Kernel.send(Process.whereis(RanMonitor.Nokia.Airscale.Manager), :get_airscale_config)`
4. **Storage Path Issues**
   - Check `priv/airscale_configs/` directory exists
   - Verify RAN Monitor has write permissions
   - Review Application Logs for filesystem errors

Download Returns 404 Error

Symptoms:

- Clicking Download shows "Configuration file not found"
- Browser shows 404 error

Possible Causes:

1. **File Path Mismatch**
   - Directory names are sanitized (lowercase, special chars replaced)

- Verify actual directory name in `priv/airscale_configs/`

2. **File Deleted**
   - Check if file was manually deleted from filesystem
   - Refresh Config Archive page to update list
3. **Permission Issues**
   - Verify web server process can read config files
   - Check file permissions on config directory

Resolution:

```
# Check directory exists
ls -la priv/airscale_configs/

# Verify file exists
ls -la priv/airscale_configs/<device-name>/

# Fix permissions if needed
chmod 755 priv/airscale_configs/
chmod 644 priv/airscale_configs/*/*.xml
```

Config Not Updating

Symptoms:

- Only `current.xml` exists, no new versions
- Version count stays at 1 even after changes

Possible Causes:

1. **Config Hasn't Changed**
   - System only creates versions when content changes
   - Review logs: `Configuration unchanged, no new version created`
2. **Polling Not Running**

- Check Application Logs for polling messages
- Verify Manager process is running
- Check for errors during config retrieval

3. **Change Detection Too Strict**
   - Whitespace-only changes are ignored (by design)
   - Verify actual parameter values changed

Verification:

```
# Check logs for config polling
grep "process_configuration" <log-file>

# Manually trigger config pull
# In IEx console:
Kernel.send(Process.whereis(RanMonitor.Nokia.Airscale.Manager),
:get_airscale_config)
```

Best Practices

Regular Backups

Recommendation: Create external backups of critical configs

Automated Backup Script Example:

```
#!/bin/bash
# backup-configs.sh - Daily config backup to external storage

BACKUP_DIR="/backup/ran-monitor/configs"
CONFIG_DIR="priv/airscale_configs"
DATE=$(date +%Y%m%d)

# Create dated backup directory
mkdir -p "$BACKUP_DIR/$DATE"

# Copy all configs
rsync -av "$CONFIG_DIR/" "$BACKUP_DIR/$DATE/"

# Keep last 30 days
find "$BACKUP_DIR" -type d -mtime +30 -exec rm -rf {} +

echo "Backup completed: $BACKUP_DIR/$DATE"
```

Schedule with cron:

```
0 2 * * * /path/to/backup-configs.sh
```

Change Documentation

Best Practice: Document why configs changed

Suggested Process:

1. Before making config changes, document the reason
2. Create a change log file alongside configs
3. Include: Date, Device, Changes, Justification, Approver

Example Change Log:

```
# config_changes.log

2025-12-30 14:35:22 - ONS-Lab-Airscale
Changed By: John Smith
Reason: Increase cell power to improve coverage in Sector 1
Parameters: txPower changed from 40dBm to 43dBm
Validated: Yes
Activated: 2025-12-30 14:40:00
Result: Coverage improved, no degradation observed

2025-12-28 09:10:45 - ONS-Lab-Airscale
Changed By: Jane Doe
Reason: Update neighbor cell list after new site deployment
Parameters: Added neighbor cells 10, 11, 12
Validated: Yes
Activated: 2025-12-28 09:15:00
Result: Handovers working correctly
```

Storage Monitoring

Recommendation: Monitor disk usage periodically

Check Storage Usage:

```
# Total size of config archive
du -sh priv/airscale_configs/

# Size per device
du -sh priv/airscale_configs/*/

# Number of versions per device
find priv/airscale_configs/ -name "*.xml" | \
  sed 's|/[^/]*\.xml||' | uniq -c
```

Set Up Alerts:

- Alert if total size exceeds threshold (e.g., 500MB)
- Alert if any device has unusually high version count
- Alert if disk space below 10% free

Integration with Configuration Management

The Config Archive system works alongside the Configuration Management page:

Workflow Integration

Download Current Config:

- Use Config Archive to get `current.xml`
- Or use Configuration Management "Download" button (triggers immediate pull)

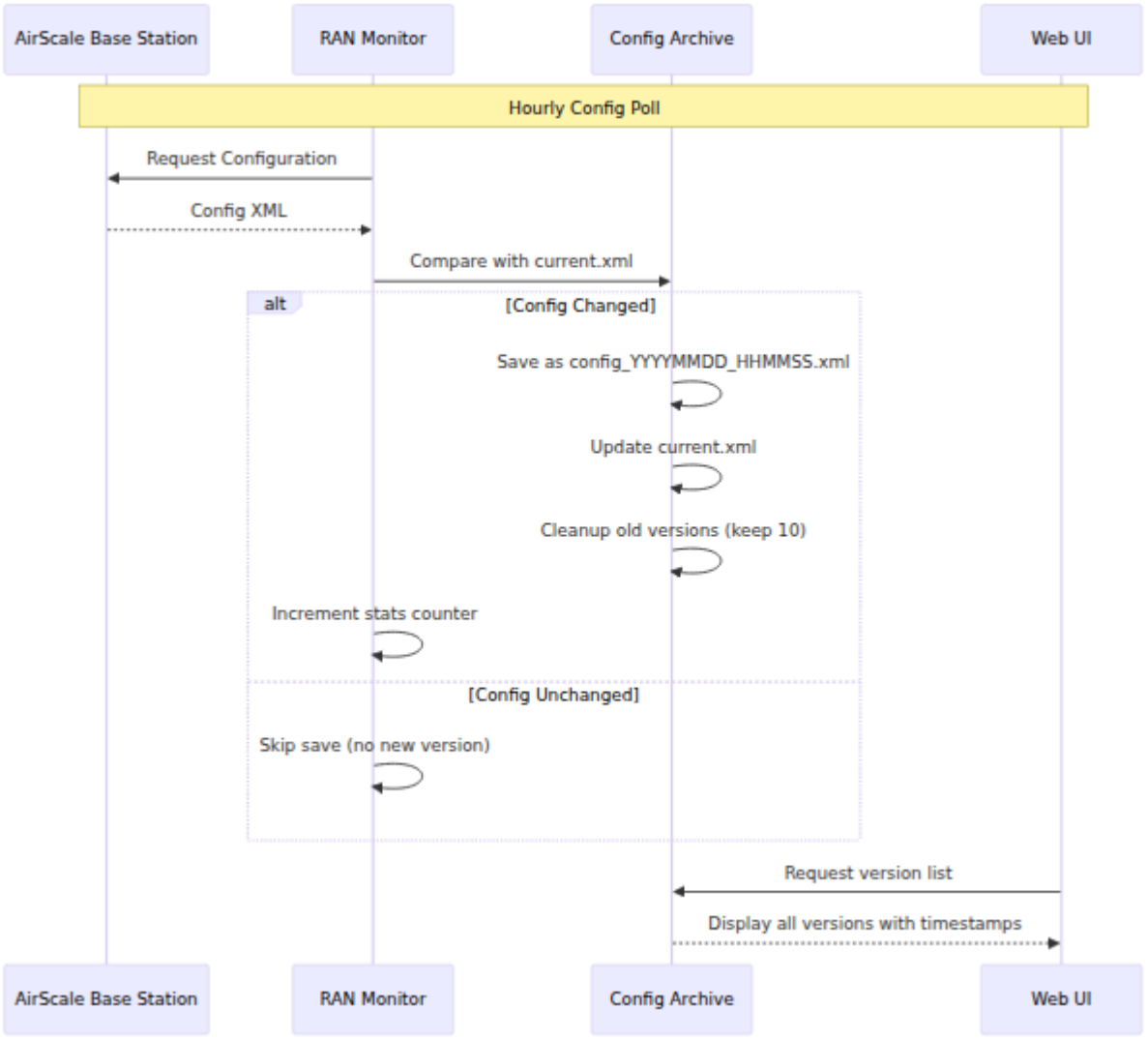
Upload Modified Config:

- Use Configuration Management page
- Upload → Validate → Activate workflow
- New version appears in Archive after next poll

Rollback Process:

- Download previous version from Archive
- Upload via Configuration Management
- Follow Validate → Activate workflow

Data Flow



API Access

While the Config Archive is primarily accessed via the Web UI, configurations can also be downloaded via direct HTTP requests.

Download Endpoints

Current Config:

```
curl -k "https://<server>:9443/download/config/<airscale-
name>/current.xml" \
  -o current_config.xml
```

Specific Version:

```
curl -k "https://<server>:9443/download/config/<airscale-
name>/ONS-Lab-Airscale_config_20251230_143522.xml" \
  -o ONS-Lab-Airscale_config_20251230_143522.xml
```

Note: AirScale name in URL must match the sanitized directory name (lowercase, underscores for special chars)

Programmatic Access

List Versions (from IEx console):

```
# Get all versions for a device
RanMonitor.Nokia.Airscale.ConfigStorage.list_config_versions("ONS-
Lab-Airscale")

# Get current config content
{:ok, xml} =
RanMonitor.Nokia.Airscale.ConfigStorage.get_current_config("ONS-
Lab-Airscale")

# Check version count
RanMonitor.Nokia.Airscale.ConfigStorage.count_versions("ONS-Lab-
Airscale")
# Returns: 655

# Get max storage size setting
RanMonitor.Nokia.Airscale.ConfigStorage.max_storage_bytes()
# Returns: 104857600 (100 MB in bytes)

# Get current storage usage
RanMonitor.Nokia.Airscale.ConfigStorage.get_storage_usage("ONS-
Lab-Airscale")
# Returns: 99614055 (bytes)

# Get detailed storage statistics
RanMonitor.Nokia.Airscale.ConfigStorage.get_storage_stats("ONS-
Lab-Airscale")
# Returns: %{version_count: 655, total_size_bytes: 99614055, ...}

# Manually cleanup (keep under 50 MB)
RanMonitor.Nokia.Airscale.ConfigStorage.cleanup_old_versions("ONS-
Lab-Airscale", 50 * 1024 * 1024)
# Returns: {:ok, 345, 50000000} - deleted 345 versions, freed 50MB

# Cleanup using default (100 MB)
RanMonitor.Nokia.Airscale.ConfigStorage.cleanup_old_versions("ONS-
Lab-Airscale")
# Returns: {:ok, 0, 0} - no cleanup needed if under limit
```

See Also

- **Web UI Guide** - Complete control panel reference
- **AirScale Configuration Guide** - Base station setup
- **Common Operations Guide** - Day-to-day management tasks
- **Data Retention Policy Guide** - Storage management

Data Retention Policy Guide

Overview

The RAN Monitor application now includes a comprehensive **Data Retention Policy** system that allows you to manage how long performance metrics, configuration data, and alarm records are stored in InfluxDB. This guide covers everything you need to know about managing data retention.

□ Quick Start

Accessing the Retention Policy Dashboard

1. Navigate to the **Control Panel**: `https://localhost:9443`
2. Click on **Data Retention** in the navigation menu
3. View and manage retention settings for all configured eNodeBs

Setting a Custom Retention Period

1. Find the eNodeB in the list
2. Update the "Retention Period" field (in hours)
3. The setting saves immediately
4. Falls back to global default if left empty

Cleaning Old Data

1. Click **Clean Old Data** button to remove records older than the retention period
2. Or click **Clear All Data** to delete all records for that eNodeB (use with caution!)

Screenshot

The Data Retention dashboard showing retention settings and record counts for each eNodeB

Features

Global Retention Settings

- **Default Retention Period**: 720 hours (30 days)
- **Configurable**: Change in `config/config.exs`
- **Fallback**: Applied to all eNodeBs without custom settings

Per-eNodeB Retention

- **Override Global**: Set custom retention for specific eNodeBs
- **Database Stored**: Persisted in the `airscales` table

- **Real-Time**: Takes effect immediately

Automatic Cleanup

- **Scheduled**: Runs every hour automatically
- **Background Worker**: `RanMonitor.Data.RetentionCleanupWorker`
- **Per-eNodeB**: Respects individual retention settings
- **Logged**: All cleanups are logged for audit trail

Data Visibility

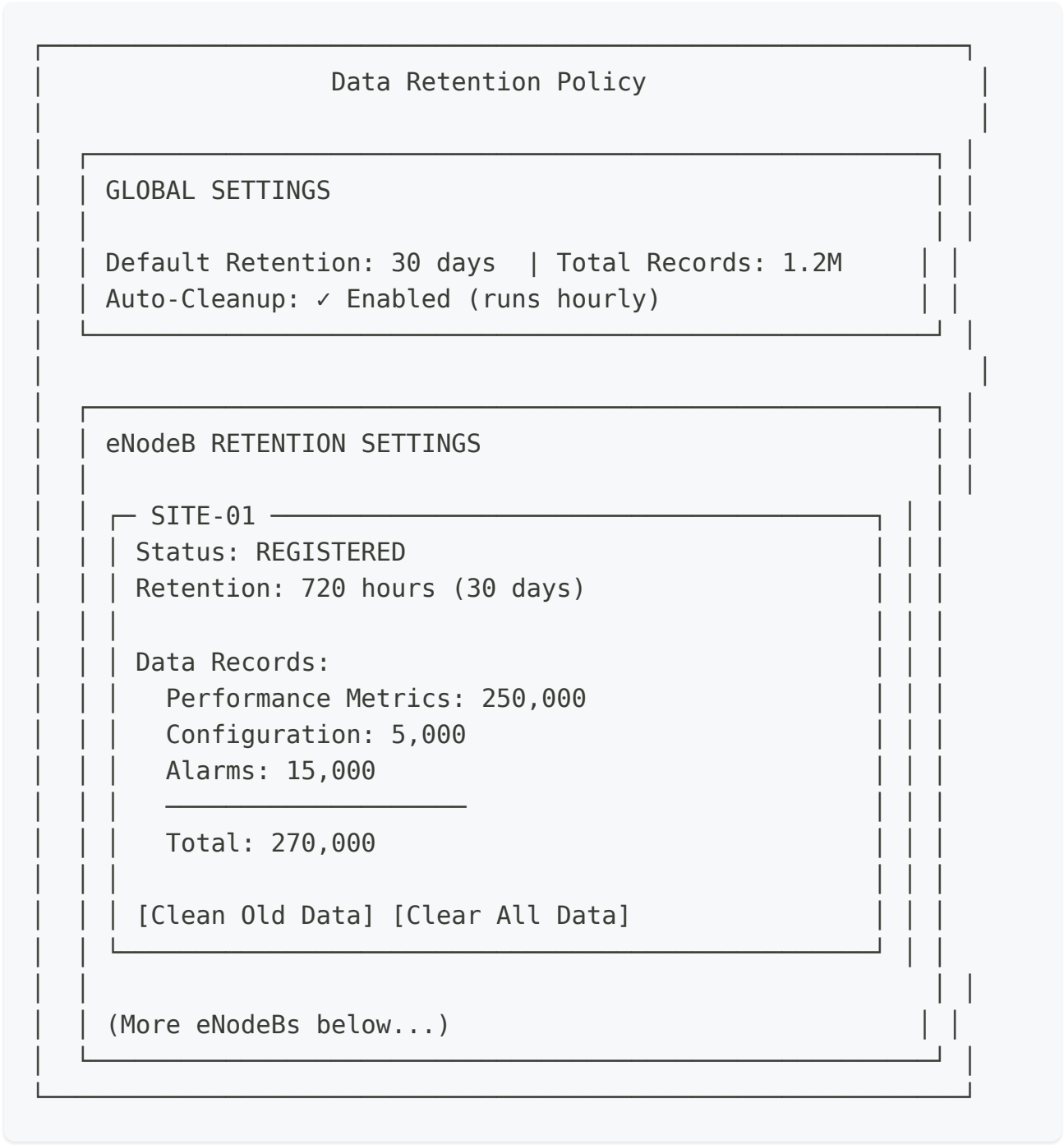
- **Record Counts**: See how many records per measurement type:
  - Performance Metrics
  - Configuration
  - Alarms
- **Total Summary**: View total records per eNodeB
- **Real-Time**: Updated on page refresh

User Interface

Dashboard Layout

The Data Retention dashboard showing global settings, per-eNodeB retention periods, and record counts

Layout Overview:



Status Indicators

- **Green (✓)**: eNodeB registered and active
- **Red (✗)**: eNodeB pending or unregistered
- **Disabled**: Cannot modify settings for non-registered eNodeBs

Action Buttons

Button	Action	Effect
Clean Old Data	Remove old records	Deletes records older than retention period
Clear All Data	Complete wipe	Deletes ALL records (⚠ use with caution!)
Refresh	Update display	Re-fetches record counts and settings

⚙ Configuration

Global Retention Configuration

Edit `config/config.exs`:

```
config :ran_monitor,
  ecto_repos: [RanMonitor.Repo],
  generators: [context_app: :ran_monitor],
  data_retention_hours: 720  # 30 days, adjust as needed
```

Supported Time Values

Period	Hours	Days	Recommended For
1 hour	1	0.04	Testing only
1 day	24	1	Short-term metrics
7 days	168	7	Weekly reports
14 days	336	14	Bi-weekly reports
30 days	720	30	Monthly reports (default)
90 days	2160	90	Long-term trends
180 days	4320	180	Bi-annual reports
1 year	8760	365	Annual reports

Environment Variables

Optionally override at runtime:

```
export DATA_RETENTION_HOURS=1440  # 60 days
mix phx.server
```

How It Works

Data Retention Flow

```
1. DATA INSERTION
   ├─ Performance Metrics → InfluxDB
   ├─ Configuration Data → InfluxDB
   └─ Alarms → InfluxDB

2. AUTOMATIC CLEANUP (Hourly)
   ├─ RetentionCleanupWorker triggers
   ├─ For each eNodeB:
   │  ├─ Get effective retention (per-eNodeB or global)
   │  ├─ Calculate cutoff timestamp
   │  └─ Delete records older than cutoff
   └─ Log results

3. MANUAL CLEANUP (On Demand)
   ├─ User clicks button in UI
   ├─ Retention policy applied
   ├─ Records deleted immediately
   └─ Success/error notification shown

4. MONITORING
   └─ Record counts displayed in UI
```

Retention Logic

Per-eNodeB Effective Retention:

```
effective_retention = case airscale.retention_hours do
  nil -> Config.data_retention_hours()      # Use global (720h)
  hours -> hours                             # Use per-eNodeB
custom value
end
```

Example:

- Global default: 720 hours (30 days)

- eNodeB "SITE-01" custom: 168 hours (7 days)
- eNodeB "SITE-02" custom: nil → uses global 720 hours

Cleanup Process

```
Cutoff Timestamp = Now - (retention_hours * 3600 seconds)

Example with 30-day retention:
├─ Current: 2025-12-11 10:00:00
├─ Retention: 720 hours (30 days)
├─ Cutoff: 2025-11-11 10:00:00
└─ Delete all records with timestamp < cutoff
```

□ Monitoring & Logging

Log Entries

The system logs all retention activities. Look for:

```
[RetentionCleanupWorker] Starting retention cleanup worker
[RetentionCleanupWorker] Cleaning data for SITE-01 (retention:
720h)
[RetentionCleanupWorker] Deleted 15,000 records for SITE-01
[RetentionCleanupWorker] Cleanup cycle complete: 5 successful, 0
failed, 75,000 total deleted
```

Monitoring Record Counts

Real-time Visibility:

1. Open Data Retention dashboard
2. View current record counts per measurement per eNodeB
3. Click "Refresh" to update counts

Historical Tracking:

- Check application logs for cleanup summaries
- Monitor InfluxDB disk usage over time
- Set up alerts based on record count growth

Advanced Usage

Programmatic Access

Use the retention policy service in your code:

```
alias RanMonitor.Data.RetentionPolicy
alias RanMonitor.Database.Nokia

# Get effective retention for an eNodeB
airscale = Nokia.get_airscale!(1)
hours = RetentionPolicy.get_retention_hours(airscale)
# => 720 (or custom value if set)

# Get record counts for an eNodeB
counts = RetentionPolicy.get_record_counts("SITE-01")
# => %{"PerformanceMetrics" => 250000, "Configuration" => 5000,
"Alarms" => 15000}

# Get total records
total = RetentionPolicy.get_total_record_count("SITE-01")
# => 270000

# Delete old records manually
{:ok, deleted_count} = RetentionPolicy.delete_old_records("SITE-
01", 720)
# => {:ok, 50000}  (50k records deleted)

# Clear all records for an eNodeB
{:ok, deleted_count} = RetentionPolicy.clear_all_records("SITE-
01")
# => {:ok, 270000}  (all 270k records deleted)
```

Adjusting Cleanup Interval

Edit `lib/ran_monitor/data/retention_cleanup_worker.ex`:

```
# Change from 1 hour (3600000ms) to 30 minutes (1800000ms)
@cleanup_interval_ms 1800000  # 30 minutes
```

Then recompile:

```
mix compile
```

Database-Level Queries

View retention settings directly:

```
SELECT name, retention_hours FROM airscales;
```

Update retention via database:

```
UPDATE airscales
SET retention_hours = 168
WHERE name = 'SITE-01';
```

Best Practices

Retention Period Selection

Short-term (< 7 days)

- Use for: Testing, staging environments
- Not recommended for: Production
- Risk: May delete important historical data

Standard (7-30 days)

- Use for: Production deployments with typical storage
- Best for: Most use cases
- Balance: Good history with manageable storage

Long-term (> 30 days)

- Use for: Trend analysis, compliance requirements
- Cost: Higher storage requirements
- Benefit: Extended historical data

Recommended by Use Case

Use Case	Retention	Reason
Daily reports	7-14 days	Weekly review cycles
Weekly reports	30-60 days	Monthly summaries
Monthly reports	90 days	Quarterly analysis
Trend analysis	180-365 days	Long-term patterns
Compliance	As required	Legal/regulatory

Storage Considerations

Estimate storage needs:

- 1000 records ≈ 1-5 KB (depending on measurement type)
- 1 million records ≈ 1-5 GB
- Retention period × collection rate = total storage

Monitor growth with:

```
# Check InfluxDB bucket size
influx bucket list

# Or check disk usage
df -h /path/to/influxdb/data
```

Security & Compliance

Data Privacy

- **No encryption** at rest by default
- **Network access** controlled via InfluxDB security
- **Access logs** available in application logs

Compliance

- **Audit trail**: All cleanups logged with timestamp
- **Data integrity**: Soft deletes, no hard wipes at application level
- **Retention proof**: Logs show what was retained/deleted

Recommendations

1. **Enable InfluxDB authentication** for production
2. **Monitor cleanup logs** regularly
3. **Set retention carefully** to balance compliance and storage
4. **Backup before bulk operations** if critical data
5. **Test retention policies** in staging first

Troubleshooting

Issue: Cleanup Not Running

Symptoms:

- Records older than retention period still exist
- No cleanup log entries

Solutions:

1. Check application is running: `ps aux | grep mix`
2. Verify `RetentionCleanupWorker` started:
   - Check logs for `[RetentionCleanupWorker] Starting`
3. Check InfluxDB connection:
   - Visit InfluxDB Status page: `https://localhost:9443/nokia/influx`
4. Verify retention settings are configured:
   - Check `config/config.exs` for `data_retention_hours`

Issue: Manual Cleanup Failed

Symptoms:

- Error message when clicking "Clean Old Data"
- Records not deleted

Solutions:

1. Check InfluxDB is accessible:
   - Test connection in dashboard
2. Verify record counts are accurate:
   - Click "Refresh" to update
3. Check application logs for errors:
   - Look for `[RetentionPolicy]` error entries
4. Verify eNodeB is registered:
   - Check eNodeB Status page

Issue: High Memory Usage After Cleanup

Symptoms:

- Application becomes slow after cleanup
- Memory usage spikes

Solutions:

1. This is normal for large deletes
2. Allow 5-10 minutes for memory to normalize
3. Consider reducing cleanup frequency:
    - Change `@cleanup_interval_ms` (default 1 hour)
4. Or reduce retention period for affected eNodeBs

Issue: Incorrect Record Counts

Symptoms:

- Record counts don't match InfluxDB UI
- "Clean Old Data" shows different numbers

Solutions:

1. Click "Refresh" to force update
2. Check InfluxDB query:
    - May take time to reflect recent deletes
3. Wait a minute and try again:
    - InfluxDB may be processing delete operations
4. Check eNodeB name matches exactly:
    - Case-sensitive comparison

Related Documentation

- **Operations Guide** - Complete operational overview

- **Web UI Guide** - Control panel reference and features
- **Getting Started Guide** - Quick start guide
- **Common Operations Guide** - Day-to-day management tasks
- **Grafana Integration Guide** - Analytics and dashboards

Access Points

- **Data Retention Dashboard**: `https://localhost:9443/nokia/retention`
- **eNodeB Status**: `https://localhost:9443/nokia/status`
- **InfluxDB Status**: `https://localhost:9443/nokia/influx`
- **Live Logs**: `https://localhost:9443/nokia/logs`

FAQ

Q: Will cleanup delete active data?

A: No. Only records older than the retention period are deleted. Currently being collected data is never affected.

Q: Can I set different retention for different eNodeBs?

A: Yes! Each eNodeB can have its own retention setting. If not set, it uses the global default.

Q: How often does automatic cleanup run?

A: Every hour by default. Adjust `@cleanup_interval_ms` in the worker if needed.

Q: What happens if I clear all data?

A: All records (Performance Metrics, Configuration, Alarms) for that eNodeB are permanently deleted. This cannot be undone.

Q: Can cleanup affect data collection?

A: No. Cleanup and data collection are independent. New data will continue being written while old data is deleted.

Q: How long does cleanup take?

A: Depends on record count:

- Small (< 100k): < 1 second
- Medium (100k-1M): 1-10 seconds
- Large (> 1M): 10-60+ seconds

Q: Can I manually delete specific records?

A: Not via the UI. Only full cleanup or complete clear available. For granular deletions, use InfluxDB CLI or API directly.

Q: What if InfluxDB is unavailable?

A: Cleanup will fail silently and retry next hour. Data collection continues unaffected.

Q: Does cleanup affect performance?

A: Minor impact during cleanup (seconds to minutes depending on data size). Hourly interval chosen to minimize impact.

🔧 Implementation Details

Files Modified

File	Changes
`lib/ran_monitor/database/nokia/airscale.ex`	Added `retention_hours` field
`lib/ran_monitor/config/config.ex`	Added `data_retention_hours()` getter
`config/config.exs`	Added global retention config and page route
`lib/ran_monitor/application.ex`	Added cleanup worker to supervision tree

Files Created

File
`lib/ran_monitor/data/retention_policy.ex`
`lib/ran_monitor/data/retention_cleanup_worker.ex`
`lib/ran_monitor/web/live/retention_policy_live.ex`
`priv/repo/migrations/20251211065257_add_retention_hours_to_airscales.`

Key Functions

RetentionPolicy Module:

- `get_retention_hours(airscale)` - Get effective retention
- `get_record_counts(airscale_name)` - Fetch record counts
- `get_total_record_count(airscale_name)` - Total count
- `delete_old_records(name, hours)` - Clean old records
- `clear_all_records(name)` - Complete wipe

RetentionCleanupWorker GenServer:

- `start_link(opts)` - Start cleanup worker
- `init(:ok)` - Initialize worker
- `handle_info(:cleanup, state)` - Run cleanup cycle

🚀 Getting Started

Setup (One-Time)

1. **Run the migration:**

   ```
   mix ecto.migrate
   ```

2. **Restart the application:**

   ```
   mix phx.server
   ```

3. **Verify installation:**

   - Navigate to `https://localhost:9443/nokia/retention`
   - Should see Data Retention dashboard

First Use

1. **Check current settings:**
   - View global retention (default: 720 hours)
   - View per-eNodeB retention settings
2. **Customize if needed:**
   - Update global retention in `config/config.exs`
   - Or set per-eNodeB via UI
3. **Monitor cleanup:**
   - Watch logs for `[RetentionCleanupWorker]` entries
   - Verify record counts decrease over time

Support

Need Help?

1. **Check logs:** Look for `[RetentionPolicy]` or `[RetentionCleanupWorker]` entries
2. **Review this guide:** Most issues covered in Troubleshooting section
3. **Check other docs:** Refer to related documentation links above
4. **Verify setup:** Ensure migration ran and worker started

Reporting Issues

Include:

- Error message from UI or logs
- eNodeB name affected
- Current retention settings
- Record counts before/after

- Steps to reproduce

Learning Resources

Related Concepts

- **InfluxDB v2.x**: Time-series database with retention policies
- **Retention Policy**: How long data is kept
- **Cleanup**: Automated deletion of old data
- **Measurement Types**: Performance Metrics, Configuration, Alarms

External Resources

- InfluxDB Documentation
- Elixir GenServer Guide
- Phoenix LiveView

Getting Started with RAN Monitor

Quick Start Guide for Deploying and Configuring RAN Monitor

Step-by-step instructions for setting up RAN Monitor in your environment

---

Table of Contents

Overview

This guide walks you through the initial deployment of RAN Monitor, from infrastructure preparation to first base station connection.

What You'll Accomplish

By the end of this guide, you will have:

- ✓ Prepared the required infrastructure (MySQL, InfluxDB)
- ✓ Configured RAN Monitor with your environment details
- ✓ Started the RAN Monitor application
- ✓ Connected your first Nokia AirScale base station
- ✓ Verified metrics are flowing to InfluxDB
- ✓ Accessed the Web UI dashboard

Estimated Time: 30-60 minutes for first-time setup

Prerequisites

Before deploying RAN Monitor, ensure you have the following:

Infrastructure Requirements

MySQL Database Server

- Version: MySQL 5.7+ or MariaDB 10.3+
- Access: Network connectivity from RAN Monitor server
- Permissions: CREATE, SELECT, INSERT, UPDATE, DELETE privileges
- Database: Empty database created for RAN Monitor
- Recommendation: Dedicated database instance or schema

InfluxDB Time-Series Database

- Version: InfluxDB 1.8+ or 2.0+
- Access: Network connectivity from RAN Monitor server
- Bucket/Database: Created and ready for metrics storage
- API Token: With write permissions to the bucket (InfluxDB 2.x)
- Storage: Sufficient disk space for your retention policy

RAN Monitor Server

- OS: Linux (Ubuntu 20.04+, CentOS 8+, or similar)
- RAM: 4GB minimum, 8GB recommended
- CPU: 2 cores minimum, 4+ cores recommended
- Disk: 20GB minimum for application and logs
- Network: Connectivity to base stations, MySQL, and InfluxDB

Network Requirements

Network Connectivity

- RAN Monitor → Nokia AirScale base stations (port 8080)
- Nokia base stations → RAN Monitor (port 9076 for webhooks)
- RAN Monitor → MySQL (port 3306)
- RAN Monitor → InfluxDB (port 8086)
- Operators → RAN Monitor Web UI (port 9443)

Firewall Rules

- Allow inbound on port 8080 (base station communication)
- Allow inbound on port 9076 (webhook receiver)
- Allow inbound on port 9443 (HTTPS Web UI)
- Allow outbound to MySQL and InfluxDB

Nokia Base Station Requirements

For Each Base Station:

- **IP Address** - Network address where base station is reachable
- **Port** - Management interface port (typically 8080)
- **Credentials** - Username and password for WebLM authentication
- **Network Route** - Verified connectivity (ping should succeed)
- **Management Interface** - Enabled and accessible

Manager Authentication Keys

- **Private Key** - For manager authentication (PEM format)
- **Public Certificate** - Manager identity certificate (DER format)
- Provided by Nokia or generated with OpenSSL

Grafana (Optional but Recommended)

- Version: Grafana 8.0+
- Access: Network connectivity to InfluxDB
- Purpose: Analytics dashboards and alerting

Initial Setup Process

Step 1: Prepare Infrastructure

1.1 Set up MySQL Database

Create the database for RAN Monitor:

```
CREATE DATABASE ran_monitor CHARACTER SET utf8mb4 COLLATE
utf8mb4_unicode_ci;
```

Create a dedicated user with appropriate privileges:

```
CREATE USER 'ran_monitor_user'@'%' IDENTIFIED BY
'secure_password';
GRANT CREATE, SELECT, INSERT, UPDATE, DELETE ON ran_monitor.* TO
'ran_monitor_user'@'%';
FLUSH PRIVILEGES;
```

Verify connectivity from RAN Monitor server:

```
mysql -h <mysql-host> -u ran_monitor_user -p ran_monitor
```

1.2 Deploy InfluxDB

For InfluxDB 1.x, create the database:

```
influx -execute 'CREATE DATABASE "nokia-monitor"'
```

For InfluxDB 2.x, create a bucket:

```
influx bucket create -n nokia-monitor -o your-org
```

Create an API token with write permissions (InfluxDB 2.x):

```
influx auth create --org your-org --write-buckets
```

Save the token for use in configuration.

1.3 Verify Network Routes

Ensure network connectivity to all base stations:

```
# Test connectivity to each base station
ping 10.7.15.66

# Verify management port is accessible
telnet 10.7.15.66 8080
```

Verify MySQL and InfluxDB are accessible:

```
# Test MySQL connectivity
telnet <mysql-host> 3306

# Test InfluxDB connectivity
curl http://<influxdb-host>:8086/ping
```

Step 2: Configure RAN Monitor

All configuration is managed in the `config/runtime.exs` file.

2.1 Database Configuration

Edit `config/runtime.exs` and configure MySQL connection:

```
config :ran_monitor, RanMonitor.Repo,
  username: "ran_monitor_user",
  password: "secure_password",
  hostname: "mysql-host",
  database: "ran_monitor",
  stacktrace: true,
  show_sensitive_data_on_connection_error: true,
  pool_size: 10
```

2.2 InfluxDB Configuration

Configure InfluxDB connection:

```
config :ran_monitor, RanMonitor.InfluxDbConnection,
  auth: [
    username: "monitor",
    password: "influx_password"  # Or API token for InfluxDB 2.x
  ],
  database: "nokia-monitor",
  host: "influxdb-host"
```

2.3 Web Endpoints Configuration

Configure the web endpoints:

```
# Main SOAP/API endpoint for base stations
config :ran_monitor, RanMonitor.Web.Endpoint,
  http: [ip: {0, 0, 0, 0}, port: 8080],
  check_origin: false,
  secret_key_base: "generate_with_mix_phx_gen_secret",
  server: true

# Control Panel Web UI (HTTPS)
config :control_panel, ControlPanelWeb.Endpoint,
  url: [host: "0.0.0.0", port: 9443, scheme: "https"],
  https: [
    ip: {0, 0, 0, 0},
    port: 9443,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ]

# Webhook endpoint for base station notifications
config :ran_monitor, RanMonitor.Web.Nokia.Airscale.Endpoint,
  url: [host: "0.0.0.0"],
  http: [ip: {0, 0, 0, 0}, port: 9076],
  server: true
```

2.4 Nokia Configuration

Configure your network identifiers and base stations:

```
config :ran_monitor,
  general: %{
    mcc: "001",  # Your Mobile Country Code
    mnc: "001"   # Your Mobile Network Code
  },
  nokia: %{
    ne3s: %{
      webhook_url: "http://<ran-monitor-ip>:9076/webhook",
      private_key: Path.join(Application.app_dir(:ran_monitor,
"priv"), "external/nokia/ne.key.pem"),
      public_key: Path.join(Application.app_dir(:ran_monitor,
"priv"), "external/nokia/ne.cert.der"),
      reregister_interval: 30
    },
    airscales: [
      %{
        address: "10.7.15.66",
        name: "Site-A-BS1",
        port: "8080",
        web_username: "admin",
        web_password: "password"
      }
    ]
  }
```

2.5 Generate SSL Certificates (if needed)

For the HTTPS Web UI, generate SSL certificates:

```
# Self-signed certificate for lab/testing
openssl req -newkey rsa:2048 -nodes -keyout
priv/cert/omnitouch.pem \
  -x509 -days 365 -out priv/cert/omnitouch.crt
```

For production, use CA-signed certificates.

For detailed configuration options, see the Runtime Configuration Guide.

Step 3: Start the System

Once configured, start RAN Monitor.

3.1 Run Database Migrations

Initialize the database schema:

```
mix ecto.migrate
```

This creates all necessary tables for session state management.

3.2 Start RAN Monitor

Start the application:

```
mix phx.server
```

Or for production deployment:

```
MIX_ENV=prod mix release
_build/prod/rel/ran_monitor/bin/ran_monitor start
```

3.3 Monitor Startup Logs

Watch the logs for successful startup:

```
[info] Running RanMonitor.Web.Endpoint with cowboy
[info] Running ControlPanelWeb.Endpoint with cowboy
[info] Running RanMonitor.Web.Nokia.Airscale.Endpoint with cowboy
[info] Starting RAN Monitor Manager
[info] Connecting to InfluxDB...
[info] InfluxDB connection established
[info] Attempting registration with device: Site-A-BS1
[info] Successfully registered with Site-A-BS1
```

Look for:

- Web endpoints started
- Database connections established
- InfluxDB connectivity confirmed
- Base station registration attempts

Verification

Step 4: Verify Operation

Check that the system is running properly.

4.1 Access the Web UI Dashboard

Open your browser and navigate to:

```
https://<ran-monitor-ip>:9443
```

You should see the RAN Monitor control panel.

4.2 Verify Base Station Status

In the Web UI:

1. Navigate to **Base Stations** page
2. Verify your base station appears in the list
3. Status should show as "Associated" (green)
4. Registration state should be "Registered"
5. Session information should show active session with expiry time

If status is red/failed, check:

- Network connectivity to base station
- Credentials are correct
- Base station management interface is accessible
- Application logs for error messages

4.3 Confirm Metrics are Flowing to InfluxDB

In the Web UI:

1. Navigate to **InfluxDB Status** page
2. Connection status should be green
3. Measurement counts should be increasing
4. Verify "Performance Metrics", "Configuration", and "Alarms" counts

Alternatively, query InfluxDB directly:

```
# InfluxDB 1.x
influx -database 'nokia-monitor' -execute 'SELECT COUNT(*) FROM
PerformanceMetrics'

# InfluxDB 2.x
influx query 'from(bucket:"nokia-monitor")
  |> range(start: -1h)
  |> filter(fn: (r) => r._measurement == "PerformanceMetrics")
  |> count()'
```

4.4 Review Startup Logs

Check application logs for any errors:

In the Web UI:

1. Navigate to **Application Logs** page
2. Filter for "Error" level
3. Verify no critical errors

Or check console output if running via `mix phx.server`.

4.5 Check Device Details

In the Web UI:

1. Click on your base station from the Base Stations page
2. Verify:
    - Registration details are populated

- Session has valid expiry time
- Recent metrics show data
- Configuration state shows parameters

Next Steps

Now that RAN Monitor is running, here are recommended next steps:

Immediate Actions

1. **Add More Base Stations**
   - Add additional devices to `config/runtime.exs`
   - Restart application to pick up changes
   - See Common Operations Guide
2. **Set Up Grafana Dashboards**
   - Install Grafana if not already deployed
   - Configure InfluxDB data source
   - Import or create dashboards
   - See Grafana Integration Guide
3. **Configure Data Retention**
   - Set appropriate retention periods
   - Configure per-device retention if needed
   - See Data Retention Policy Guide
4. **Configure Alarms and Alerts**
   - Review active alarms in Web UI
   - Set up Grafana alert rules
   - Configure notification channels
   - See Alarm Management Guide

Operational Readiness

Documentation Review:

- Read Web UI Guide for daily operations
- Review Common Operations Guide for routine tasks
- Study Troubleshooting Guide for issue resolution

Team Training:

- Walk through Web UI with operations team
- Practice common workflows (daily health check, alarm investigation)
- Review escalation procedures for critical alarms

Monitoring Setup:

- Create operational dashboards in Grafana
- Set up alert rules for critical metrics
- Configure notification channels (Slack, email, PagerDuty)

Security Hardening:

- Replace self-signed certificates with CA-signed certificates
- Move credentials to environment variables
- Restrict file permissions on `config/runtime.exs`
- Configure firewall rules

Production Deployment

Before Production:

- Test in staging environment first
- Verify all base stations connect successfully
- Confirm metrics are accurate
- Test alarm notifications
- Document any custom configuration

Production Launch:

- Deploy during maintenance window
- Monitor closely for first 24 hours
- Have rollback plan ready
- Keep support contacts available

Ongoing Operations:

- Daily health checks via Web UI
- Weekly review of alarm trends
- Monthly capacity planning with Grafana
- Regular configuration backups

Getting Help

Troubleshooting Resources

- Troubleshooting Guide - Common issues and solutions
- Web UI Guide - Control panel reference
- Application Logs page - Real-time system logs

Documentation

- Operations Guide - Complete operational reference
- Runtime Configuration Guide - Configuration details
- AirScale Configuration - Base station setup

Common First-Time Issues

Base Station Not Registering:

- Verify network connectivity (ping)
- Check credentials are correct

- Confirm port 8080 is accessible
- Review application logs for errors

InfluxDB Connection Failed:

- Verify InfluxDB is running
- Check host and port configuration
- Confirm API token has write permissions
- Test connectivity: `curl http://<influxdb-host>:8086/ping`

Web UI Not Accessible:

- Verify HTTPS port 9443 is open
- Check SSL certificates are present
- Confirm web endpoint started in logs
- Try accessing from local machine first

Related Documentation

- **Operations Guide** - Complete operational overview
- **Web UI Guide** - Control panel user guide
- **Common Operations Guide** - Day-to-day tasks
- **Runtime Configuration Guide** - Configuration reference
- **AirScale Configuration** - Base station setup
- **Grafana Integration Guide** - Analytics and dashboards
- **Alarm Management Guide** - Alarm handling
- **Data Retention Policy Guide** - Data management
- **Troubleshooting Guide** - Problem resolution

Grafana Integration & Analytics Guide

Building Operational Dashboards and Alerts for RAN Monitoring

Complete guide to Grafana dashboarding, alerting strategies, and KPI visualization

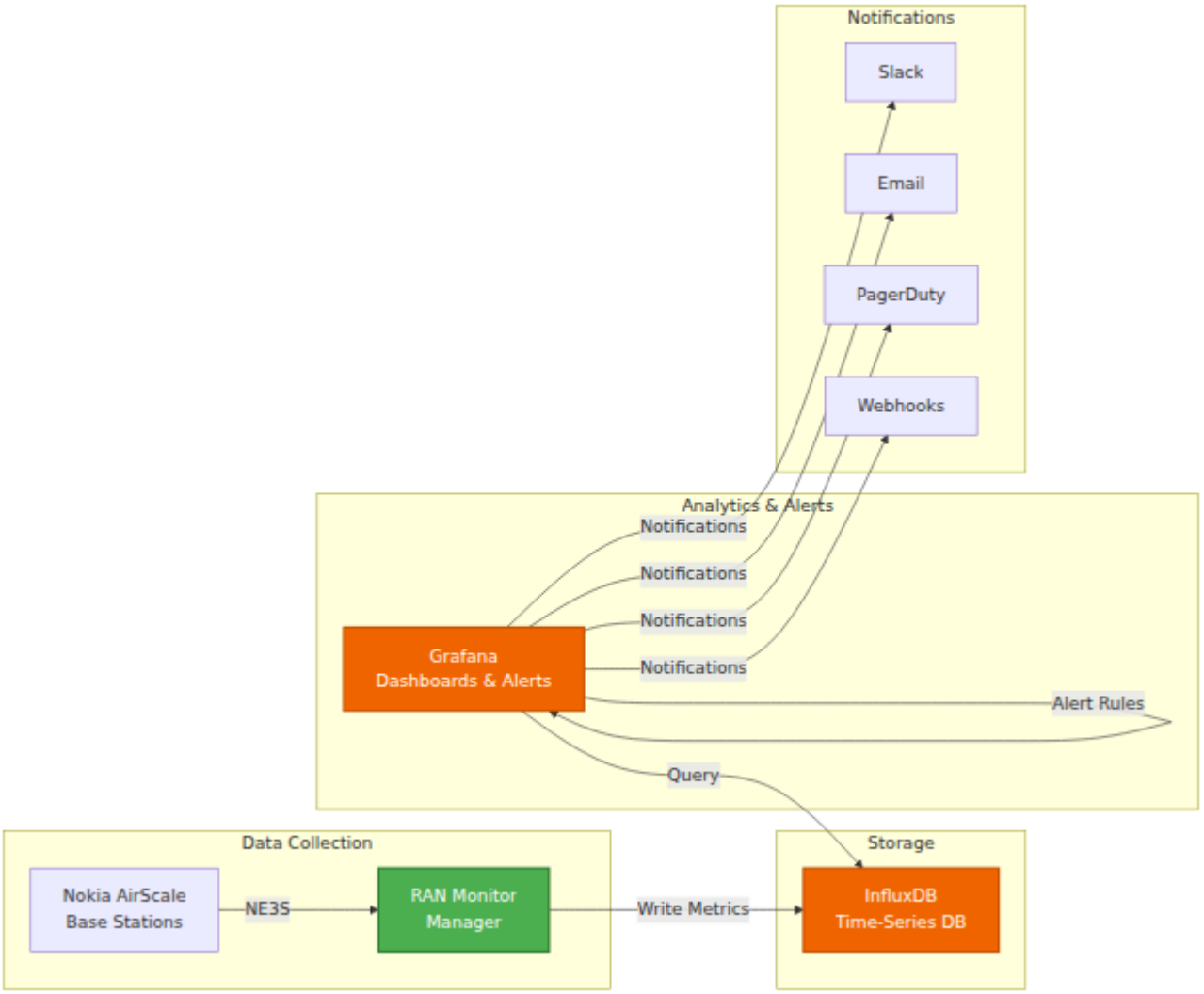
---

Table of Contents

Overview

Grafana is a visualization and alerting platform that transforms the metrics collected by RAN Monitor into actionable insights for network operations teams.

Monitoring Architecture



Grafana Benefits

- **Real-Time Visibility** - Live dashboards showing current network state
- **Historical Analysis** - Trend analysis over days/weeks/months
- **Alerting** - Proactive notifications before issues impact users
- **Custom Views** - Dashboards tailored to different roles (exec, ops, engineering)
- **Reporting** - Snapshot exports and scheduled reports

Dashboard Customization

Important: The dashboards and visualizations described in this guide are **examples and templates**. The **Operations/NOC (ONS) team** will design

and build Grafana dashboards according to their specific operational requirements, KPIs, and monitoring workflows.

This guide provides:

- Query examples and patterns to build upon
- Best practices for dashboard organization
- Alert configuration templates
- Counter reference mappings (see Nokia Counter Reference)

The ONS team should customize:

- Panel layouts and visualizations
- Alert thresholds and escalation policies
- Retention policies for their data volume (see Data Retention Policy)
- Aggregation windows based on monitoring needs
- Notification channels and routing

For runtime configuration options and data collection settings, refer to the Runtime Configuration Guide.

Grafana & InfluxDB Setup

Installation

Prerequisites:

- InfluxDB 2.0+ with bucket created for RAN Monitor
- InfluxDB API token with read permissions
- Network connectivity between Grafana and InfluxDB

Docker Compose Example:

```yaml
version: '3.8'
services:
  influxdb:
    image: influxdb:2.7
    environment:
      INFLUXDB_DB: ran_metrics
      INFLUXDB_ADMIN_USER: admin
      INFLUXDB_ADMIN_PASSWORD: change_me
    ports:
      - "8086:8086"
    volumes:
      - influxdb_data:/var/lib/influxdb2

  grafana:
    image: grafana/grafana:latest
    environment:
      GF_SECURITY_ADMIN_PASSWORD: change_me
    ports:
      - "3000:3000"
    depends_on:
      - influxdb
    volumes:
      - grafana_data:/var/lib/grafana
      - ./provisioning:/etc/grafana/provisioning

volumes:
  influxdb_data:
  grafana_data:
```

Creating an InfluxDB API Token

1. Open InfluxDB UI (port 8086)
2. Navigate to API Tokens
3. Create new token with permissions:
   - Read: buckets, `ran_metrics` (your bucket)
4. Copy token value
5. Use in Grafana data source configuration

Data Source Configuration

Adding InfluxDB as Data Source in Grafana

1. **Access Data Sources**
   - Grafana → Configuration → Data Sources
2. **Create New Data Source**
   - Click "Add data source"
   - Select "InfluxDB"
3. **Configure Connection**

Setting	Value	Notes
Name	RAN Monitor	Display name in Grafana
URL	`http://influxdb:8086`	Must be reachable from Grafana
Access	Server (default)	Grafana backend accesses DB
Organization	omnitouch	Your InfluxDB org
Token	(API token)	From API token creation
Default Bucket	ran_metrics	Where RAN Monitor writes
Min time interval	10s	Matches polling interval

4. **Test Connection**
   - Click "Test" button

- Should show "DataSource is working"

Note: The InfluxDB connection settings (URL, organization, bucket name) must match your RAN Monitor configuration. See Runtime Configuration Guide for details on InfluxDB setup and AirScale Configuration for base station registration.

Flux Query Language

Grafana uses Flux to query InfluxDB. Basic syntax:

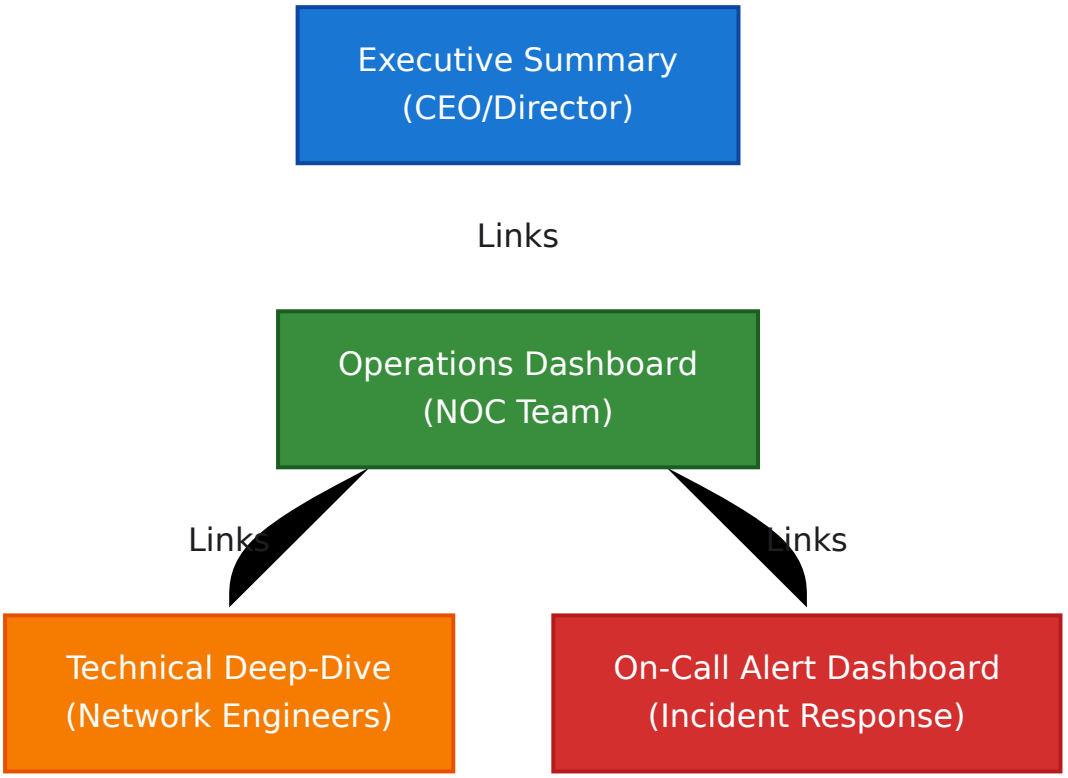
```
from(bucket:"ran_metrics")
  |> range(start: -7d, stop: now())
  |> filter(fn: (r) => r._measurement == "PerformanceMetrics")
  |> filter(fn: (r) => r.device == "SITE_A_BS1")
  |> group(by: ["_field"])
  |> aggregateWindow(every: 1h, fn: mean)
```

Key Concepts:

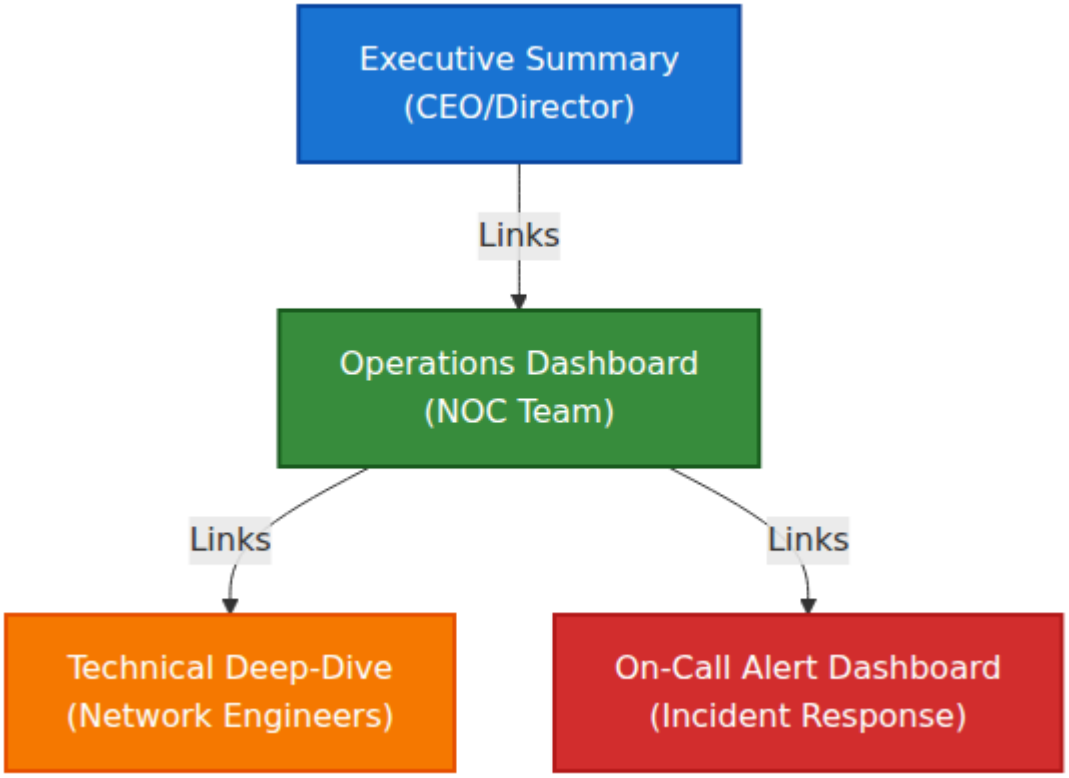
- `from()` - Select bucket
- `range()` - Time window
- `filter()` - Select data
- `group()` - Organize results
- `aggregateWindow()` - Summarize time periods

Dashboard Design Patterns

Dashboard Hierarchy



Panel Types & Use Cases



Standard Dashboard Sections

Top Section: Key Metrics (Status Indicators)

Show current state at a glance:

```
┌─────────────────────────────────────────────────┐
│ Network Health Snapshot                         │
├─────────────────┬──────────────────┬────────────┤
│ Devices Up      │ Active Alarms    │ Avg Cell Avail  │
│ 48/50 (96%)     │ 3 Critical       │ 98.5%           │
├─────────────────┴──────────────────┴────────────┤
│ Most Recent Incident: [2 hours ago] Fixed       │
└─────────────────────────────────────────────────┘
```

Purpose:

- Quick status verification (< 10 seconds to assess)
- Green/red indicators for immediate issues

- Links to detailed dashboards for investigation

Middle Section: Trends (Time Series Charts)

Show patterns and changes over time:

```
┌──────────────────────────────────────────────────────┐
│ Traffic Patterns (7 days)                            │
│ [Large area chart with daily/weekly patterns]        │
│ Peak: 250 Gbps (Wednesday 2pm)                       │
│ Valley: 80 Gbps (Sunday 3am)                         │
└──────────────────────────────────────────────────────┘
```

Purpose:

- Identify capacity constraints
- Understand traffic patterns
- Predict peak times
- Detect anomalies

Bottom Section: Details & Alerts (Tables)

Show granular information:

```
┌──────────────────────────────────────────────────────┐
│ Active Alarms (Sorted by Severity)                   │
├─────────┬──────────────┬──────────────┬──────────────┤
│ Level   │ Device       │ Issue        │ Duration     │
├─────────┼──────────────┼──────────────┼──────────────┤
│ 🔴      │ SITE_A_BS1   │ Cell Down    │ 45 minutes   │
│ 🟡      │ SITE_B_BS2   │ High Temp    │ 2 hours      │
└──────────────────────────────────────────────────────┘
```

Purpose:

- Immediate action items
- Investigation details
- Trend information (duration, frequency)

Query Examples

Note: The following query examples use Nokia-specific performance counters. For detailed counter definitions, units, and usage guidelines, refer to the Nokia Counter Reference. For configuring data collection intervals and InfluxDB settings, see the Runtime Configuration Guide.

Performance Metrics Queries

Cell Availability by Device (Last 24 Hours)

```
from(bucket:"ran_metrics")
  |> range(start: -24h)
  |> filter(fn: (r) => r._measurement == "PerformanceMetrics")
  |> filter(fn: (r) => r._field == "CellAvailability")
  |> group(by: ["device"])
  |> aggregateWindow(every: 1h, fn: mean)
  |> yield(name: "cell_availability")
```

Usage:

- Executive dashboard for SLA reporting
- Time series chart showing hourly averages
- Target: > 99.5% availability

Traffic Throughput Trend (7 Days)

```
from(bucket:"ran_metrics")
  |> range(start: -7d)
  |> filter(fn: (r) => r._measurement == "PerformanceMetrics")
  |> filter(fn: (r) => r._field =~ /Throughput.*/)
  |> group(by: ["device", "_field"])
  |> aggregateWindow(every: 10m, fn: mean)
  |> yield(name: "traffic_trend")
```

Usage:

- Capacity planning dashboard
- Area chart showing peak vs. valley
- Identify busy hours for scheduling

DL Resource Utilization by Cell

```
from(bucket:"ran_metrics")
  |> range(start: -1h)
  |> filter(fn: (r) => r._measurement == "PerformanceMetrics")
  |> filter(fn: (r) => r._field == "DLResourceUtilization")
  |> filter(fn: (r) => r.device == "SITE_A_BS1")
  |> aggregateWindow(every: 10s, fn: last)
  |> yield(name: "dl_resource")
```

Usage:

- Real-time operations dashboard
- Gauge panel warning at 80%, critical at 95%
- Quick identification of congested cells

Alarm Queries

Active Alarms by Severity (Last 24 Hours)

```
from(bucket:"ran_metrics")
  |> range(start: -24h)
  |> filter(fn: (r) => r._measurement == "Alarms")
  |> filter(fn: (r) => r.status == "active")
  |> group(by: ["severity"])
  |> count()
  |> yield(name: "alarm_count")
```

Usage:

- Status indicator showing alarm counts
- Pie chart of distribution
- Click-through to detailed alarm list

Alarm Rate (Alarms per Hour)

```
from(bucket:"ran_metrics")
  |> range(start: -7d)
  |> filter(fn: (r) => r._measurement == "Alarms")
  |> group(by: ["severity"])
  |> aggregateWindow(every: 1h, fn: count)
  |> yield(name: "alarm_rate")
```

Usage:

- Trend chart showing when alarm storms occur
- Identify times of high instability
- Correlate with configuration changes

Frequently Triggered Alarms

```
from(bucket:"ran_metrics")
  |> range(start: -7d)
  |> filter(fn: (r) => r._measurement == "Alarms")
  |> group(by: ["alarm_description"])
  |> count()
  |> sort(columns: ["_value"], desc: true)
  |> limit(n: 10)
  |> yield(name: "top_alarms")
```

Usage:

- Identify systemic issues
- Prioritize engineering efforts
- Root cause analysis focus

Advanced Analytics

Cell Availability Forecast (Linear Regression)

```
from(bucket:"ran_metrics")
  |> range(start: -30d)
  |> filter(fn: (r) => r._measurement == "PerformanceMetrics")
  |> filter(fn: (r) => r._field == "CellAvailability")
  |> filter(fn: (r) => r.device == "SITE_A_BS1")
  |> aggregateWindow(every: 1h, fn: mean)
  |> statefulWindow(every: 1h, period: 24h)
    |> map(fn: (r) => ({r with _value: float(v: r._value)}))
    |> reduce(fn: (r, acc) => ({
        x: acc.x + [float(v: r._time)],
        y: acc.y + [r._value]
      }),
      initial: {x: [], y: []})
    |> yield(name: "availability_forecast")
```

Usage:

- Predict when SLA may be breached
- Proactive maintenance scheduling
- Capacity forecasting

Handover Success Correlation with Traffic

```
from(bucket:"ran_metrics")
  |> range(start: -7d)
  |> filter(fn: (r) => r._measurement == "PerformanceMetrics")
  |> filter(fn: (r) => r._field =~ /HandoverSuccess|Traffic/)
  |> group(by: ["device", "_field"])
  |> aggregateWindow(every: 1h, fn: mean)
  |> pivot(rowKey: ["_time"], columnKey: ["_field"], valueColumn:
"_value")
  |> map(fn: (r) => ({r with correlation: float(v:
r.HandoverSuccess) * float(v: r.Traffic)}))
  |> yield(name: "ho_traffic_correlation")
```

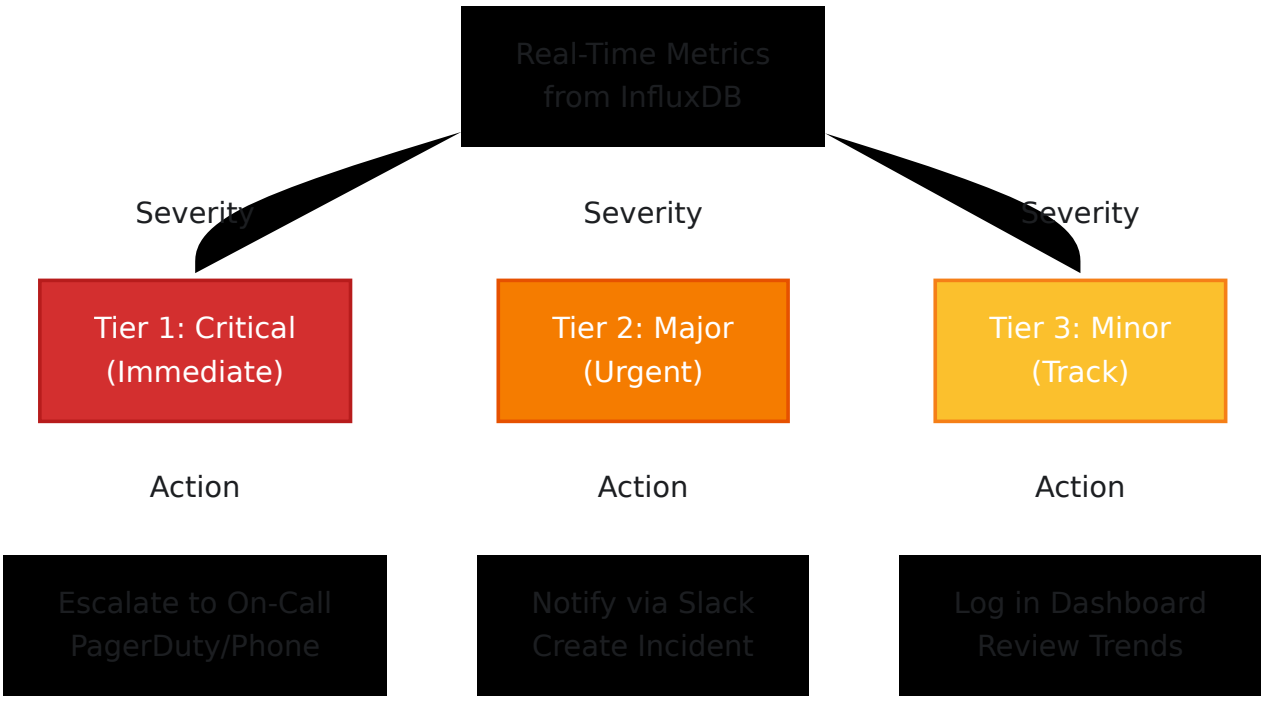
Usage:

- Identify if handover issues are load-related
- Tune handover hysteresis thresholds

- Network optimization insights

Alert Rules & Escalation

Alert Strategy Framework



Creating Alert Rules in Grafana

Step 1: Create Alert Rule

1. Open Dashboard
2. Click panel to alert on
3. Panel → Create alert
4. Or Alerting → Alert rules → Create new alert rule

Step 2: Configure Evaluation Criteria

Example 1: Cell Availability Alert

```
Condition: CellAvailability < 95%
Duration: 15 minutes
Evaluation Frequency: Every 1 minute
For: The last 15 minutes
```

Rationale:

- Trigger at 95% to warn before SLA breach (99.5%)
- 15-minute window to avoid false positives from transients
- Monitor every minute for rapid response

Example 2: Alarm Storm Detection

```
Condition: count(active_alarms) > 10
Duration: 5 minutes
Evaluation Frequency: Every 2 minutes
For: The last 5 minutes
```

Rationale:

- 10+ alarms indicate systemic issue
- Quick 5-minute detection for rapid response
- Check frequently to catch escalation

Example 3: DL Resource Exhaustion

```
Condition: DLResourceUtilization > 90%
Duration: 30 minutes
Evaluation Frequency: Every 5 minutes
For: The last 30 minutes
```

Rationale:

- Sustained high resource usage indicates congestion
- 30-minute window prevents false alerts from traffic spikes
- Monitor every 5 minutes to catch sustained congestion

Step 3: Configure Notification Channel

1. Click "Notification channel"
2. Select or create channel (Slack, Email, PagerDuty, etc.)
3. Configure message template

Message Template Example:

```
Alert: {{ .AlertRuleName }}
Severity: {{ .Severity }}
Device: {{ .Labels.device }}
Value: {{ .EvalMatches[0].Value }}
Duration: {{ .StartsAt }}

{{ .RuleUrl }}
```

Escalation Policies

Tier 1 Alerts (Critical):

- **Condition:** Service impacting (device down, SLA breach imminent)
- **Duration:** Immediate (1-5 minutes)
- **Notification:** Phone call + SMS + Slack + PagerDuty
- **Owner:** On-call engineer
- **SLA:** Response in < 15 minutes

Tier 2 Alerts (Major):

- **Condition:** Degraded performance (quality, availability trending down)
- **Duration:** 15-30 minutes
- **Notification:** Slack + Email + PagerDuty
- **Owner:** NOC team + senior engineer
- **SLA:** Response in < 30 minutes

Tier 3 Alerts (Minor):

- **Condition:** Informational (trends, approaching limits)
- **Duration:** 1+ hours

- **Notification:** Slack + Dashboard
- **Owner:** Capacity planning / engineering
- **SLA:** Daily review

Notification Channels

Slack Integration

```
1. Create Slack App in workspace
2. Get webhook URL
3. In Grafana Alerting → Notification channels
4. Add "Slack" channel
5. Paste webhook URL
6. Test notification
```

Slack Message Formatting:

```
 CRITICAL: Cell Down - SITE_A_BS1_Cell1
Duration: 45 minutes
Impact: ~2000 subscribers
Last successful data: 2:15 PM

[Investigate] [Acknowledge] [Dashboard]
```

PagerDuty Integration

```
1. Create integration key in PagerDuty
2. In Grafana Alerting → Notification channels
3. Add "PagerDuty" channel
4. Paste integration key
5. Map severity levels:
   - Critical → Trigger incident
   - Major → Trigger with lower urgency
   - Minor → Add to existing incident
```

Email Integration

```
1. Configure SMTP in Grafana config
2. In Alerting → Notification channels
3. Add "Email" channel
4. Enter recipient addresses
5. Can include CSV of recipients for distribution lists
```

Operational Dashboards

Dashboard 1: Executive Health Dashboard

Audience: Management, executives **Refresh Rate:** 5 minutes **Purpose:** High-level health overview

Panels:

1. **Status Summary (4 Stat Panels)**
    - Devices Up / Total
    - Active Alarms (color-coded by severity)
    - Avg Cell Availability (%)
    - Current Peak Traffic (Gbps)
2. **Network Health (Time Series)**
    - Cell Availability trend (7 days)
    - Alarm rate trend (7 days)
    - Traffic forecast vs. actual
3. **Recent Incidents (Table)**
    - Time, Duration, Root Cause, Status
    - Last 7 days, sorted by severity
4. **Device Status Grid (Heatmap)**
    - Rows: Devices, Columns: Health metrics

- Green (OK) → Yellow (Degraded) → Red (Down)

Example Dashboard:

Example showing base station overview with Last Reported, Connected UEs, Data Transferred, PRB Utilization, and Throughput metrics.

Dashboard 2: NOC Operations Dashboard

Audience: Network operations center team **Refresh Rate:** 10 seconds
Purpose: Real-time operational control

Panels:

1. **Active Issues (Table)**
   - Time, Severity, Device, Issue, Duration
   - Sort by severity, click to drill-down
2. **Resource Utilization (Gauges)**
   - DL Resource % (per site)
   - UL Resource % (per site)
   - CPU % on devices

3. **Traffic Overview (Area Chart)**
    - DL/UL throughput (last 24 hours)
    - Current vs. 24-hour average
    - Peak hour indicators
4. **Alarm Trend (Bar Chart)**
    - Count by severity (last hour, rolling)
    - Stacked bar showing distribution
5. **Device Status (Quick View)**
    - Device name, IP, Status (green/red)
    - Last metric update timestamp
    - Links to device-specific dashboard
6. **Recent Events (Time Series)**
    - Alarms appearing/clearing
    - Configuration changes
    - Session status changes

Example Dashboard:

Example showing 4G Status overview with geographic map, alarm table with severity levels, and performance statistics.

Dashboard 3: Engineering Deep-Dive

Audience: Network engineers **Refresh Rate:** 1 minute **Purpose:** Detailed technical analysis

Panels:

1. **Traffic Pattern Analysis (Multi-Series)**
    - DL/UL by site for comparison
    - Baseline + current overlay
    - Hour-of-day seasonality
2. **Cell Quality Metrics (Multi-Series)**
    - SINR distribution (histogram)
    - RSRP distribution (histogram)
    - Handover success rate trends
3. **Radio Performance (Time Series)**
    - RLC retransmission rate (by site)
    - RRC setup success rate
    - Call drop rate
4. **Configuration Audit (Table)**
    - Device, Config Date, Parameters Changed
    - Highlights recent modifications
5. **Correlation Analysis (Scatter)**
    - DL Resource vs. Traffic
    - Traffic vs. Handover Success
    - Availability vs. Alarm Count

Dashboard 4: On-Call Alert Dashboard

Audience: Incident responders (on-call) **Refresh Rate:** 5 seconds **Purpose:** Rapid incident assessment and response

Panels:

1. **Alert Summary (Large Stat)**
    - Count of active critical alerts
    - Background color: Green (OK) / Red (Issues)
2. **Critical Issues (Table, Big Text)**
    - Device, Issue, Duration
    - Auto-scroll to newest first
3. **Related Metrics (Time Series)**
    - Traffic, Resource utilization
    - Quality metrics for affected device
    - Auto-populate based on alert
4. **Recent Changes (Table)**
    - Configuration changes in last 4 hours
    - Software versions
    - Parameter modifications
5. **Similar Issues (Table)**
    - Same issue type in last 30 days
    - Time to resolution
    - Root causes identified
6. **Escalation Path (Text Panel)**
    - Next level escalation contact
    - Maintenance window info
    - Related ticket/incident number

Dashboard 5: Nokia AirScale Detailed Performance

Audience: RF engineers, performance analysts **Refresh Rate:** 30 seconds
Purpose: Deep-dive Nokia-specific metrics and KPIs

This dashboard uses Nokia-specific performance counters to provide comprehensive visibility into AirScale base station performance. See the Nokia Counter Reference for detailed counter definitions.

Panel 1: Resource Utilization Overview (Gauges)

Shows current PRB (Physical Resource Block) usage:

```
// Downlink PRB Usage
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8011C37")
  |> filter(fn: (r) => r._field == "counterValue")
  |> group()
  |> mean()
  |> map(fn: (r) => ({ r with _value: r._value / 10.0}))  //
Convert to percentage
  |> rename(columns: {_value: "DL PRB Usage"})

// Uplink PRB Usage
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8011C24")
  |> filter(fn: (r) => r._field == "counterValue")
  |> group()
  |> mean()
  |> map(fn: (r) => ({ r with _value: r._value / 10.0}))  //
Convert to percentage
  |> rename(columns: {_value: "UL PRB Usage"})
```

Visualization: Gauge panels with thresholds:

- Green: 0-70%
- Yellow: 70-85%
- Red: 85-100%

Panel 2: Throughput Trends (Time Series)

Displays PDCP layer throughput for downlink and uplink:

```
// Downlink Throughput
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8012C26")
  |> filter(fn: (r) => r._field == "counterValue")
  |> map(fn: (r) => ({ r with _value: r._value / 1000.0}))  //
Convert to Mbps
  |> rename(columns: {_value: "Downlink Mbps"})

// Uplink Throughput
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8012C23")
  |> filter(fn: (r) => r._field == "counterValue")
  |> map(fn: (r) => ({ r with _value: r._value / 1000.0}))  //
Convert to Mbps
  |> rename(columns: {_value: "Uplink Mbps"})
```

Counters Used:

- **M8012C26** - PDCP Throughput DL Mean (kbit/s)
- **M8012C23** - PDCP Throughput UL Mean (kbit/s)

Panel 3: Active UE Count (Time Series)

Tracks the number of connected users:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8018C1")
  |> filter(fn: (r) => r._field == "counterValue")
  |> rename(columns: {_value: "UEs Connected"})
```

Counter Used:

- **M8018C1** - Active UE per eNB max (count)

Panel 4: Cell Availability (Time Series with Alert Threshold)

Calculates and displays cell availability percentage:

```
import "strings"

from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8020C3" or
                       r["metricCounter"] == "M8020C6" or
                       r["metricCounter"] == "M8020C4")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> pivot(rowKey:["_time"], columnKey: ["metricCounter"],
valueColumn: "_value")
  |> map(fn: (r) => ({
      _time: r._time,
      "Cell Availability": 100.0 * r.M8020C3 / (r.M8020C6 -
r.M8020C4)
    }))
```

Counters Used:

- **M8020C3** - Samples when the cell is available

- **M8020C6** - Cell availability denominator
- **M8020C4** - Samples when the cell is planned unavailable

Threshold Alert: Cell Availability < 99%

Panel 5: Per-Cell PRB Utilization (Multi-Series Time Series)

Shows resource usage broken down by individual cells:

```
import "strings"

// Uplink PRB per cell
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8011C24")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> map(fn: (r) => ({ r with _value: r._value / 10.0}))
  |> rename(columns: {"_value": "Average Uplink PRB Usage"})

// Downlink PRB per cell
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8011C37")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> map(fn: (r) => ({ r with _value: r._value / 10.0 }))
  |> rename(columns: {"_value": "Average Downlink PRB Usage"})
```

Panel 6: Per-Cell Throughput (Multi-Series Time Series)

PDCP throughput broken down by cell:

```
import "strings"

// Downlink PDCP Throughput per cell
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8012C26")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> rename(columns: {"_value": "Downlink PDCP Throughput"})

// Uplink PDCP Throughput per cell
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8012C23")
  |> filter(fn: (r) => r._field == "counterValue")
  |> group()
  |> rename(columns: {"_value": "Uplink PDCP Throughput"})
```

Panel 7: RSSI (Received Signal Strength Indicator) (Multi-Series Time Series)

Displays uplink signal strength statistics:

```
import "strings"

// Minimum RSSI
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8005C0")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> rename(columns: {"_value": "Minimum RSSI"})

// Mean RSSI
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8005C2")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> rename(columns: {"_value": "Mean RSSI"})

// Maximum RSSI
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8005C1")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> rename(columns: {"_value": "Maximum RSSI"})
```

Counters Used:

- **M8005C0** - RSSI for PUCCH Min (dBm)
- **M8005C1** - RSSI for PUCCH Max (dBm)

- **M8005C2** - RSSI for PUCCH Mean (dBm)

Panel 8: Latency (Time Series)

PDCP SDU delay measurement:

```
import "strings"

from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8001C2")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> rename(columns: {"_value": "Latency"})
```

Counter Used:

- **M8001C2** - PDCP SDU delay on DL DTCH Mean (ms)

Panel 9: RRC Connection Setup Success Rate (Time Series with Threshold)

Calculates the percentage of successful connection setups:

```
import "strings"

from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M8013C5" or
                       r["metricCounter"] == "M8013C17" or
                       r["metricCounter"] == "M8013C18" or
                       r["metricCounter"] == "M8013C19" or
                       r["metricCounter"] == "M8013C34" or
                       r["metricCounter"] == "M8013C31" or
                       r["metricCounter"] == "M8013C21" or
                       r["metricCounter"] == "M8013C93" or
                       r["metricCounter"] == "M8013C91")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${CellKey}"))
  |> group()
  |> pivot(rowKey:["_time"], columnKey: ["metricCounter"],
valueColumn: "_value")
  |> map(fn: (r) => ({
      _time: r._time,
      "Setup Success Ratio": 100.0 * r.M8013C5 / (r.M8013C17 +
r.M8013C18 + r.M8013C19 + r.M8013C34 + r.M8013C31 + r.M8013C21 +
r.M8013C93 + r.M8013C91)
    }))
```

Counters Used:

- **M8013C5** - Signaling Connection Establishment Completions
- **M8013C17-M8013C93** - Various connection attempt types

Threshold Alert: Setup Success Ratio < 95%

Panel 10: VSWR (Voltage Standing Wave Ratio) by Antenna (Time Series)

Hardware health monitoring for antenna systems:

```
import "strings"

from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M40001C0")
  |> filter(fn: (r) => r._field == "counterValue")
  |> filter(fn: (r) => strings.containsStr(v: r["DN"], substr:
"${RadioKey}"))
  |> map(fn: (r) => ({
      r with
      "DN": strings.split(v: r["DN"], t: "/")[5],
      "VSWR": r._value / 10.0
    }))
  |> group()
  |> pivot(rowKey: ["_time"], columnKey: ["DN"], valueColumn:
"VSWR")
```

Counter Used:

- **M40001C0** - VSWR per antenna branch (0.1 ratio)

Threshold Alert: VSWR > 2.0

Panel 11: Power Consumption (Time Series)

Base station power usage monitoring:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["recordType"] == "performanceMetric")
  |> filter(fn: (r) => r["basebandName"] == "${Airscale}")
  |> filter(fn: (r) => r["metricCounter"] == "M40002C2")
  |> filter(fn: (r) => r._field == "counterValue")
  |> group()
  |> map(fn: (r) => ({ r with _value: r._value / 100000.0 }))
  |> rename(columns: {"_value": "Power Consumption"})
```

Counter Used:

- **M40002C2** - Power Consumption (factor of 100000)

Dashboard Variables:

This dashboard uses Grafana template variables for dynamic filtering:

- **${Airscale}** - Base station selector (dropdown)
- **${CellKey}** - Cell selector for multi-cell sites (dropdown)
- **${RadioKey}** - Radio unit selector for VSWR (dropdown)

Alert Rules for this Dashboard:

1. **High PRB Utilization** - Triggers when DL or UL PRB > 85% for 5 minutes
2. **Low Cell Availability** - Triggers when availability < 99% for 10 minutes
3. **Poor Setup Success Rate** - Triggers when RRC setup success < 95% for 5 minutes
4. **High VSWR** - Triggers when VSWR > 2.0 for any antenna for 15 minutes
5. **Abnormal Power Consumption** - Triggers when power deviates > 20% from baseline

Using this Dashboard:

1. **Capacity Analysis** - Monitor PRB utilization to identify cells approaching capacity
2. **Performance Troubleshooting** - Use RSSI, latency, and setup success rate to diagnose issues
3. **Hardware Health** - Track VSWR and power consumption for proactive maintenance
4. **Quality Assurance** - Monitor availability and throughput for SLA compliance

See Nokia Counter Reference for complete counter definitions and usage guidelines.

Example Dashboard - Detailed View:

Nokia Monitor detailed dashboard showing S1 Connections, operational state, data transferred, UEs connected, average PRB usage, performance monitoring metrics, and geographic operation map.

Example Dashboard - Cell Availability, PRB Usage, and Throughput:

Cell Availability charts per LNCEL, LTE PRB Usage per TTI for uplink/downlink, and PDCP Throughput charts showing performance across multiple cells.

Example Dashboard - RSSI, Power, Latency, and RRC:

RSSI charts (Min/Mean/Max), Combined Power Consumption, Latency measurements, RRC Setup Success Ratio, and VSWR (RMOD) charts for multiple cells.

Example Dashboard - Additional Performance Panels:

Additional performance panels showing RRC Setup Success Ratio continuation, Latency measurements, VSWR RMOD charts, and UEs Connected over time.

Example Dashboard - VSWR Detail with Tooltip:

Detailed VSWR RMOD-2 chart showing antenna measurements (ANTL-1, ANTL-2, ANTL-3, ANTL-4) with interactive tooltip displaying timestamp and values.

Troubleshooting

No Data Appearing in Panels

Symptoms:

- Dashboard loads but panels show "No data"
- Data source appears connected

Diagnosis:

```
1. Check InfluxDB query is valid
2. Verify measurement name exists in InfluxDB
3. Check time range includes data points
4. Verify filter conditions match data tags
```

Solution:

- Test query in InfluxDB UI directly
- Adjust time range (try last 24 hours)

- Check tag names match RAN Monitor output
- Enable query inspector to see actual results

Slow Dashboard Loading

Symptoms:

- Dashboard takes > 5 seconds to load
- Panels appear one by one slowly

Diagnosis:

```
1. Too many panels (> 8)
2. Queries are too complex/large data range
3. InfluxDB performance issues
4. Network latency
```

Solution:

- Reduce number of panels
- Limit time range (24h vs. 1 year)
- Pre-aggregate data in InfluxDB
- Check InfluxDB CPU/memory
- Increase query timeout

Alerts Not Firing

Symptoms:

- Alert rule is enabled
- Condition should be met
- No notifications received

Diagnosis:

```
1. Check alert evaluation is happening
2. Verify notification channel is working
3. Check alert rule condition
4. Review notification channel logs
```

Solution:

- Test alert rule manually (pencil icon → Test)
- Check alert rule status in Alerting → Alert rules
- Verify notification channel has correct URL/key
- Check Grafana logs for errors
- Re-test notification channel with manual trigger

Incorrect Data in Dashboards

Symptoms:

- Values don't match expected numbers
- Data looks shifted in time
- Aggregations seem wrong

Diagnosis:

```
1. Check time zone settings
2. Verify aggregation function
3. Check tag/label filters
4. Review query for math errors
```

Solution:

- Set dashboard time zone to match InfluxDB
- Verify aggregateWindow function (mean/sum/last)
- Test filters in InfluxDB directly
- Simplify query to isolate issue

Data Retention Issues

Symptoms:

- Historical data missing or incomplete
- Queries return less data than expected
- Dashboard shows gaps in time series

Solution:

- Check retention policy settings in Data Retention Policy
- Verify retention period is sufficient for your query time range
- Adjust per-eNodeB retention if needed

Configuration Issues

Symptoms:

- InfluxDB connection errors
- Missing eNodeB data in queries
- Incorrect data collection intervals

Solution:

- Review Runtime Configuration Guide for proper settings
- Verify AirScale Configuration is correct
- Check eNodeB registration status

Related Documentation

- **Nokia Counter Reference** - Complete performance counter definitions
- **Data Retention Policy** - Managing data lifecycle and storage
- **Runtime Configuration Guide** - System configuration and tuning
- **AirScale Configuration** - Base station setup and registration

Common Operations Guide

Day-to-Day RAN Monitor Management Tasks

Practical guide for routine operational tasks and device management

---

Table of Contents

Overview

This guide covers routine operational tasks for managing RAN Monitor in day-to-day operations. These procedures are designed for NOC teams, network administrators, and operations staff.

Prerequisites

- RAN Monitor is installed and running
- You have access to the configuration files

- You can restart the RAN Monitor application
- You understand your network topology

For initial setup, see the Getting Started Guide.

Adding a New Base Station

When deploying new Nokia AirScale base stations, follow these steps to add them to monitoring.

Step 1: Verify Network Connectivity

Before adding the device to configuration, ensure network connectivity:

```
# Test basic connectivity
ping <base-station-ip>

# Verify management port is accessible
telnet <base-station-ip> 8080
```

Expected Result: Successful ping responses and telnet connection

If Fails:

- Check network routes
- Verify firewall rules allow connectivity
- Confirm base station is powered on and operational

Step 2: Gather Device Information

Collect the following information:

Information	Example	Where to Find
IP Address	10.7.15.67	Network documentation or device label
Port	8080	Typically 8080 for Nokia AirScale
Device Name	Site-B-Tower-1	Use site naming convention
Username	admin	From base station provisioning
Password	password123	From base station provisioning

Device Naming Best Practices:

- Use descriptive, consistent naming
- Include site identifier
- Include equipment type if multiple at same site
- Examples: `NYC-SiteA-BS1`, `LAX-Tower-Main`, `CHI-Indoor-DAS`

Step 3: Check for Unconfigured Devices

Before manually adding, check if the device has already attempted to connect:

1. Open Web UI: `https://<ran-monitor-ip>:9443`
2. Navigate to **Unconfigured eNodeBs** page
3. Look for your device's IP address or Agent ID
4. Note the Agent ID if found

This helps verify the device can reach RAN Monitor.

Step 4: Add Device to Configuration

Edit `config/runtime.exs` and add the new device to the `airscales` list:

```
config :ran_monitor,
  nokia: %{
    ne3s: %{
      # ... existing ne3s configuration ...
    },
    airscales: [
      # Existing devices
      %{
        address: "10.7.15.66",
        name: "Site-A-BS1",
        port: "8080",
        web_username: "admin",
        web_password: "password1"
      },

      # New device
      %{
        address: "10.7.15.67",          # IP address of new base
station
        name: "Site-B-Tower-1",          # Descriptive name
        port: "8080",                    # Management port
        web_username: "admin",           # WebLM username
        web_password: "password123"      # WebLM password
      }
    ]
  }
```

Important: Ensure proper Elixir syntax - commas, indentation, and map structure must be correct.

Step 5: Validate Configuration

Before restarting, validate the configuration syntax:

```
elixir -c config/runtime.exs
```

Expected Output: No errors

If Errors:

- Check for missing commas
- Verify all opening braces `{` and brackets `[` are closed
- Ensure strings are properly quoted
- Check indentation is consistent

Step 6: Restart RAN Monitor

Restart the application to load the new configuration:

```
# If running in development
# Press Ctrl+C twice, then:
mix phx.server

# If running as a service
systemctl restart ran_monitor

# If running via release
/path/to/ran_monitor/bin/ran_monitor restart
```

Step 7: Verify Device Registration

After restart, verify the device is now monitored:

1. **Check Application Logs:**

   ```
   [info] Attempting registration with device: Site-B-Tower-1
   [info] Successfully registered with Site-B-Tower-1
   ```

2. **Check Web UI:**

   - Navigate to **Base Stations** page
   - Find your new device in the list
   - Status should be "Associated" (green)
   - Registration state should be "Registered"

3. **Check Device Details:**

- Click on the device
- Verify session information shows active session
- Confirm "Last Contact" timestamp is recent

4. **Check InfluxDB Status:**
   - Navigate to **InfluxDB Status** page
   - Verify total record count is increasing
   - Measurement counts should grow as data is collected

Step 8: Configure Data Retention (Optional)

If this device requires different retention than the global default:

1. Navigate to **Data Retention** page
2. Find your new device in the list
3. Update the "Retention Period" field (in hours)
4. System saves automatically

For more details, see the Data Retention Policy Guide.

Step 9: Add to Grafana Dashboards

Update Grafana dashboards to include the new device:

1. Edit dashboard template variables
2. Add device name to dropdown selectors
3. Create device-specific dashboard if needed

For more details, see the Grafana Integration Guide.

Removing a Base Station

When decommissioning a base station, follow these steps to remove it from monitoring.

Step 1: Decide on Data Handling

Before removing, decide what to do with historical data:

Option A: Preserve Data

- Disable monitoring but keep historical data
- Useful for decommissioned but potentially returning equipment

Option B: Remove Data

- Delete all historical data for the device
- Frees up InfluxDB storage
- Irreversible - data cannot be recovered

Step 2: Disable Device (Preserve Data)

To stop monitoring but keep historical data:

Edit `config/runtime.exs` and locate the device in the `airscales` list. Comment it out or remove it:

```
airscales: [
  %{
    address: "10.7.15.66",
    name: "Site-A-BS1",
    port: "8080",
    web_username: "admin",
    web_password: "password1"
  },

  # Decommissioned device - commented out to preserve data
  # %{
  #   address: "10.7.15.67",
  #   name: "Site-B-Tower-1",
  #   port: "8080",
  #   web_username: "admin",
  #   web_password: "password123"
  # }
]
```

Step 3: Remove Data (Optional)

To delete all historical data for the device:

1. Navigate to **Data Retention** page in Web UI
2. Find the device in the list
3. Click **Clear All Data** button
4. Confirm the action

Warning: This is permanent and cannot be undone.

Step 4: Restart RAN Monitor

Restart the application to apply configuration changes:

```
systemctl restart ran_monitor
# or
mix phx.server
```

Step 5: Verify Removal

After restart:

1. **Check Base Stations Page:**
    - Device should no longer appear in active list
    - If data was preserved, device may still show in historical queries
2. **Check Application Logs:**
    - No registration attempts for removed device
    - No errors about missing device
3. **Check InfluxDB:**
    - If data was deleted, record counts should be lower
    - Device should not appear in new metrics

Step 6: Update Grafana Dashboards

Remove device from Grafana configurations:

1. Edit dashboard template variables
2. Remove device name from dropdowns
3. Delete device-specific dashboards if they exist

Updating Device Credentials

When base station passwords are changed, update RAN Monitor configuration.

Step 1: Note Current Status

Before making changes:

1. Verify device is currently connected (green status)
2. Note current session information
3. Take screenshot or record current state for comparison

Step 2: Update Configuration

Edit `config/runtime.exs` and update the credentials:

```
airscales: [
  %{
    address: "10.7.15.66",
    name: "Site-A-BS1",
    port: "8080",
    web_username: "admin",
    web_password: "new_password_here"  # Updated password
  }
]
```

Step 3: Restart RAN Monitor

Apply the configuration change:

```
systemctl restart ran_monitor
```

Step 4: Verify Reconnection

After restart:

1. **Check Application Logs:**

   ```
   [info] Attempting registration with device: Site-A-BS1
   [info] Successfully registered with Site-A-BS1
   ```

2. **Check Web UI:**

   - Device status should be "Associated" (green)
   - New session should be established
   - "Last Contact" should be recent

If Device Fails to Connect:

- Verify new credentials are correct
- Check for typos in password
- Confirm credentials work directly on base station
- Review application logs for authentication errors

Adjusting Collection Intervals

Change how frequently RAN Monitor collects data from base stations.

Understanding Collection Intervals

RAN Monitor collects three types of data at different intervals:

Data Type	Default Interval	Configurable	Impact of Shorter Interval
Performance Metrics	10 seconds	Yes	More granular data, higher network/storage usage
Alarms	10 seconds	Yes	Faster alarm detection, more queries
Configuration	60 seconds	Yes	More frequent config snapshots, more storage
Health Checks	30 seconds	Yes	More responsive to connectivity issues

When to Adjust Intervals

Shorten Intervals (More Frequent):

- Troubleshooting active issues
- High-value critical infrastructure
- SLA monitoring with tight requirements
- Capacity testing and analysis

Lengthen Intervals (Less Frequent):

- Reduce network traffic
- Reduce InfluxDB storage usage
- Lower-priority test environments
- Bandwidth-constrained links

Step 1: Modify Configuration

Collection intervals are configured in the application code, not runtime.exs. To change them, you'll need to modify the source code and recompile.

Example locations (may vary by version):

- Performance metrics: `lib/ran_monitor/nokia/airscale/manager.ex`
- Alarms: `lib/ran_monitor/nokia/airscale/manager.ex`
- Configuration: `lib/ran_monitor/nokia/airscale/manager.ex`

Note: Contact Omnitouch support for assistance with collection interval changes, as this requires source code modifications.

Step 2: Consider Impact

Before changing intervals:

Network Impact:

- Calculate: devices counters interval = queries per second
- Shorter intervals = more network traffic
- Ensure network can handle increased load

Storage Impact:

- Calculate: data points per day retention period = total storage
- Example: 10s interval = 8,640 measurements/day per counter
- Ensure InfluxDB has sufficient disk space

System Performance:

- More frequent polling = higher CPU usage on RAN Monitor
- Monitor system resources after changes

Step 3: Monitor After Changes

After adjusting intervals:

1. **Watch Application Logs** for any errors
2. **Monitor System Resources:**
   - CPU usage on RAN Monitor server
   - Network bandwidth utilization
   - InfluxDB disk I/O and storage growth
3. **Verify Data Quality:**
   - Check InfluxDB for expected measurement frequency
   - Verify no gaps in data
4. **Adjust if Needed:**
   - Revert if system is overloaded
   - Fine-tune based on observed performance

Managing Device Configuration

How to safely manage base station configurations through RAN Monitor.

Configuration Workflow

For detailed configuration management procedures, see the Web UI Guide - Configuration Management.

Quick Reference:

1. **Download** current configuration (backup)
2. **Modify** configuration offline
3. **Upload** new configuration get Plan ID
4. **Validate** configuration using Plan ID
5. **Activate** if validation succeeds
6. **Verify** changes took effect

Best Practices

Always Download First:

- Keep backup of current configuration
- Enables rollback if needed
- Documents configuration before change

Validate Before Activating:

- Never activate without validating
- Validation catches syntax errors
- Prevents service interruptions

Schedule Changes Appropriately:

- Use maintenance windows when possible
- Avoid peak traffic hours
- Have rollback plan ready

Document Changes:

- Record what was changed and why
- Note Plan ID for tracking
- Document verification results
- Update change management system

Monitor After Changes:

- Watch for alarms
- Verify metrics normalize
- Check device stability for 15-30 minutes
- Be prepared to rollback if issues occur

Monitoring System Health

Routine checks to ensure RAN Monitor is operating correctly.

Daily Health Check

Perform these checks at the start of each shift:

1. Access Web UI Dashboard

```
https://<ran-monitor-ip>:9443
```

2. Review System Status

- All devices showing green (associated) status?
- Any red (failed) devices requiring investigation?
- Alarm count reasonable for time of day?

3. Check Alarm Summary

- Any critical alarms active?
- Alarm rate trending up or down?
- Any repeat alarms indicating systemic issues?

4. Verify Data Collection

- Navigate to InfluxDB Status page
- Measurement counts increasing?
- Last update timestamp recent?

5. Review Recent Logs

- Navigate to Application Logs page
- Filter for "Error" level
- Any recurring errors?

For detailed health check procedures, see the Web UI Guide - Web UI Workflows.

Example: Grafana Monitoring Dashboard

Comprehensive monitoring dashboard showing S1 connections status by LNMME, operational state, data transferred, UEs connected, average PRB usage, performance monitoring metrics, and geographic coverage map. This dashboard provides at-a-glance visibility into network health and device status.

Weekly System Review

More thorough check performed weekly:

1. Review Alarm Trends

- Use Grafana to analyze alarm rate over past week
- Identify any alarm storms or patterns
- Correlate with maintenance or changes

2. Check Storage Growth

- InfluxDB disk usage trending
- MySQL database size
- Application log file sizes

3. Review Device Connectivity

- Any devices with frequent disconnections?

- Session timeout issues?
- Pattern of connectivity problems?

4. System Resource Utilization

- CPU usage on RAN Monitor server
- Memory usage trends
- Network bandwidth consumption

5. Configuration Changes

- Review all configuration changes made
- Verify changes were documented
- Correlate with any issues

Time Required: 30-45 minutes

Data Management

Managing Data Retention

See the Data Retention Policy Guide for complete details.

Quick Reference:

View Current Retention:

- Navigate to Data Retention page
- Check global default and per-device settings

Adjust Retention Period:

- Update retention hours for specific device
- Or modify global default in `config/config.exs`

Clean Old Data:

- Manual cleanup: Click "Clean Old Data" button
- Automatic cleanup runs hourly

Storage Planning:

- Monitor InfluxDB disk usage weekly
- Adjust retention based on available storage
- Balance retention period with storage capacity

Exporting Data

Export Device Configurations:

1. Navigate to device detail page
2. Click "Download Configuration"
3. Save XML file to safe location
4. Label with device name and date

Export Metrics (via InfluxDB):

```
# Export data for specific device
influx -database 'nokia-monitor' -execute '
  SELECT * FROM PerformanceMetrics
  WHERE basebandName='''Site-A-BS1'''
  AND time > now() - 7d
' -format csv > export.csv
```

Export via Grafana:

- Open dashboard
- Select time range
- Click "Share" → "Export" → "CSV"

Backup and Recovery

Regular Backups

What to Backup:

1. Configuration Files

```
# Backup runtime configuration
cp config/runtime.exs backups/runtime.exs.$(date +%Y%m%d)

# Backup entire config directory
tar -czf backups/config-$(date +%Y%m%d).tar.gz config/
```

2. Device Configurations

- Download configurations from all devices via Web UI
- Store in version control or backup location
- Perform weekly or before/after changes

3. MySQL Database

```
# Backup session state database
mysqldump -u ran_monitor_user -p ran_monitor >
backups/ran_monitor-$(date +%Y%m%d).sql
```

4. InfluxDB Data

```
# InfluxDB 1.x backup
influxd backup -portable -database nokia-monitor
/backups/influx-$(date +%Y%m%d)

# InfluxDB 2.x backup
influx backup /backups/influx-$(date +%Y%m%d)
```

5. SSL Certificates

```
cp priv/cert/* backups/certificates-$(date +%Y%m%d)/
```

Backup Schedule

Daily:

- MySQL database (session state)
- Configuration files if changed

Weekly:

- InfluxDB data (or per retention policy)
- Device configurations from all base stations

Before Changes:

- Configuration files
- Device configurations
- Database snapshot

Recovery Process

Recover from Configuration Error:

1. Stop RAN Monitor

   ```
   systemctl stop ran_monitor
   ```

2. Restore configuration file

   ```
   cp backups/runtime.exs.20251230 config/runtime.exs
   ```

3. Validate configuration

   ```
   elixir -c config/runtime.exs
   ```

4. Restart RAN Monitor

```
systemctl start ran_monitor
```

5. Verify devices reconnect

Recover from Database Loss:

1. Stop RAN Monitor

```
systemctl stop ran_monitor
```

2. Restore MySQL database

```
mysql -u ran_monitor_user -p ran_monitor < backups/ran_monitor-
20251230.sql
```

3. Restart RAN Monitor

```
systemctl start ran_monitor
```

4. Devices will re-register automatically
5. New metrics collection begins
6. Historical data remains in InfluxDB

Recover from Complete System Loss:

1. Reinstall RAN Monitor on new server
2. Restore configuration files
3. Restore MySQL database
4. Restore InfluxDB data
5. Restore SSL certificates
6. Start RAN Monitor
7. Verify all devices reconnect

8. Verify historical data is accessible

System Maintenance

Routine Maintenance Tasks

Monthly:

1. Review Logs

- Archive old application logs
- Check for recurring errors or warnings
- Clean up log files to free disk space

2. Update Documentation

- Document any configuration changes
- Update network diagrams if topology changed
- Review and update operational procedures

3. Security Updates

- Apply OS security patches
- Update database software if needed
- Rotate passwords per security policy

4. Performance Review

- Analyze system resource trends
- Identify any performance degradation
- Plan capacity upgrades if needed

Quarterly:

1. Disaster Recovery Test

- Test backup restoration process
- Verify recovery procedures work
- Update disaster recovery documentation

2. Security Audit

- Review access logs
- Verify user permissions
- Update SSL certificates if expiring soon

3. Capacity Planning

- Review storage growth trends
- Forecast future capacity needs
- Plan hardware upgrades if needed

Annually:

1. SSL Certificate Renewal

- Replace expiring SSL certificates
- Test new certificates before expiry

2. Password Rotation

- Update all base station credentials
- Update database passwords
- Update API tokens

3. System Upgrade

- Plan RAN Monitor version upgrade
- Test in staging environment
- Schedule production upgrade

Maintenance Windows

Planning a Maintenance Window:

1. **Schedule During Low Traffic:**
    - Nights or weekends typically best
    - Avoid busy hours identified in Grafana
2. **Notify Stakeholders:**
    - Inform operations team
    - Update status page
    - Set expectations for downtime
3. **Prepare Rollback Plan:**
    - Backup current state
    - Document rollback steps
    - Have previous version ready if upgrading
4. **Perform Maintenance:**
    - Follow documented procedures
    - Monitor progress closely
    - Document any deviations
5. **Verify System Health:**
    - All devices reconnect
    - Metrics flowing normally
    - No errors in logs
    - Run health check procedures
6. **Document Results:**
    - Record what was done
    - Note any issues encountered
    - Update procedures if needed

Related Documentation

- **Getting Started Guide** - Initial setup procedures
- **Web UI Guide** - Control panel user guide
- **Runtime Configuration Guide** - Configuration reference
- **AirScale Configuration** - Base station setup
- **Grafana Integration Guide** - Analytics and dashboards
- **Alarm Management Guide** - Alarm handling procedures
- **Data Retention Policy Guide** - Data lifecycle management
- **Troubleshooting Guide** - Problem resolution
- **Operations Guide** - Complete operational overview

Nokia LTE Performance Counter Reference

Comprehensive guide to Nokia AirScale/FlexiRadio performance measurement counters

---

Table of Contents

Overview

Nokia LTE base stations (AirScale, FlexiRadio) report performance data using a structured counter system. Each counter measures a specific aspect of network performance, from resource utilization to radio quality.

What are Performance Counters?

Performance counters are numeric measurements collected by the base station at regular intervals. They provide visibility into:

- How busy the network is (resource utilization)
- How well it's performing (throughput, quality)
- How many users are connected (load)
- Whether services are available (availability)
- Signal quality and radio conditions

Measurement Groups

Counters are organized into measurement groups, each covering a specific functional area. The complete Nokia LTE counter set includes the following measurement groups:

Core Performance Measurements (M8xxx)

Measurement	Category	Counter Count	Primary Focus
M8000	S1 Interface	33	Initial Context Setup, S1 setup/reset, UE context
M8001	Cell Performance	336	PDCP delay, RACH, transport blocks, MCS distribution
M8004	X2 Interface	4	X2 data volume, inter-eNB traffic
M8005	Radio Quality	237	RSSI, SINR, radio conditions, AMC
M8006	EPS Bearer	54	Bearer setup/modification/release
M8007	Data Radio Bearer	14	DRB establishment and management
M8008	RRC Connection Reject	14	Connection rejection reasons and statistics
M8009	Handover Preparation	8	HO preparation failures
M8010	CQI Distribution	27	Channel Quality Indicator statistics
M8011	Resource Utilization	55	PRB usage, resource allocation

Measurement	Category	Counter Count	Primary Focus
M8012	Throughput	121	PDCP data rates, volume statistics
M8013	Signaling Connection	21	RRC connection setup attempts/success
M8014	Inter-eNB Handover	14	X2-based handover procedures
M8015	Intra-eNB Handover	13	Internal cell handovers
M8016	CS Fallback	18	Circuit-switched fallback procedures
M8017	Inter-System HO	10	Handover to other RATs (3G/2G)
M8018	eNB Load	8	Active UE counts, load statistics
M8019	NACC	4	Network Assisted Cell Change
M8020	Cell Availability	7	Cell state and availability sampling
M8021	Inter-Frequency HO	17	Inter-frequency handover procedures
M8022	X2 Setup	2	X2 interface establishment

Measurement	Category	Counter Count	Primary Focus
M8023	Data Volume	36	PDCP SDU volume on air interface

Network Interface Measurements (M5xxx)

Measurement	Category	Counter Count	Primary Focus
M5112	IP Interface Input	112	Incoming packet statistics, interface metrics
M5113	Ethernet RX	21	Received Ethernet packet statistics

Hardware Measurements (M4xxxx)

Measurement	Category	Counter Count	Primary Focus
M40001	Radio Hardware	Variable	VSWR, antenna health, RF metrics
M40002	Power Consumption	Variable	Base station power usage

Counter Naming Convention

Nokia counters follow a standard naming format:

```
M<measurement><type>C<counter>
```

Example: M8011C24

- **M** - Fixed prefix indicating "Measurement"
- **8011** - Measurement group (Cell Resource)
- **C** - Fixed separator indicating "Counter"
- **24** - Specific counter within the group (UL PRB utilization mean)

Logical Types

Each counter has a logical type that determines how it aggregates data:

- **Sum** - Cumulative count of events
- **Average** - Mean value over measurement period
- **Min** - Minimum value observed
- **Max** - Maximum value observed
- **Current** - Current sampled value
- **Denominator** - Used for ratio calculations

Counter Categories

By Functional Area

Capacity Planning:

- M8011Cxx - PRB utilization
- M8018Cxx - Active UE counts
- M8012Cxx - Throughput rates

Performance Monitoring:

- M8001Cxx - Latency and delay
- M8005Cxx - RSSI, SINR

- M8013Cxx - Setup success rates

Availability Tracking:

- M8020Cxx - Cell availability
- M8049Cxx - Connection status

Troubleshooting:

- M8005Cxx - Radio quality issues
- M8001Cxx - Queue problems
- M8013Cxx - Setup failures

Resource Utilization Counters

M8011 - Cell Resource Measurements

These counters track Physical Resource Block (PRB) utilization, which indicates how busy the cell's radio resources are.

Counter	Name	Description	Unit	Type	Scaling
M8011C24	UL PRB utilization per TTI Mean	Average uplink PRB usage per Transmission Time Interval	0.1%	Average	Divide by 10 for percentage
M8011C37	DL PRB utilisation per TTI Mean	Average downlink PRB usage per Transmission Time Interval	0.1%	Average	Divide by 10 for percentage

Understanding PRB Utilization:

- **Physical Resource Blocks (PRBs)** are the smallest units of radio resource allocation in LTE
- **TTI (Transmission Time Interval)** = 1 millisecond in LTE
- Higher utilization = busier cell (more traffic)
- 100% utilization indicates cell is at capacity

Value Ranges:

- 0-1000 (represents 0.0% to 100.0%)
- Divide counter value by 10 to get percentage
- Example: Counter value 453 = 45.3% PRB utilization

Usage in Capacity Planning:

- < 50% - Cell has spare capacity
- 50-70% - Normal loaded cell
- 70-85% - Heavily loaded, monitor for congestion
- > 85% - Near capacity, consider adding sectors/carriers

Grafana Query Example:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] == "M8011C37")
  |> filter(fn: (r) => r._field == "counterValue")
  |> map(fn: (r) => ({ r with _value: r._value / 10.0}))  //
Convert to percentage
```

Throughput Counters

M8012 - Cell Throughput Measurements

These counters measure PDCP (Packet Data Convergence Protocol) layer throughput, indicating actual user data transfer rates.

Counter	Name	Description	Unit	Type	Update Trigger
M8012C23	PDCP Throughput UL Mean	Mean uplink PDCP throughput	kbit/s	Average	When PDCP SDU received from UE
M8012C26	PDCP Throughput DL Mean	Mean downlink PDCP throughput	kbit/s	Average	When PDCP SDU transmitted to UE

Understanding PDCP Throughput:

- **PDCP** layer is where user data packets are processed
- Throughput is measured in kilobits per second (kbit/s)
- Represents actual data transfer rate for user traffic
- Updated every second

Throughput Calculation:

- Measured as average over 1-second sampling interval
- Accounts for all active users in the cell
- Includes both VoLTE and data bearers

Performance Benchmarks:

Downlink (M8012C26):

- < 10 Mbps - Low traffic / few users
- 10-50 Mbps - Moderate traffic
- 50-100 Mbps - High traffic / many active users
- > 100 Mbps - Peak traffic (depends on cell bandwidth)

Uplink (M8012C23):

- < 5 Mbps - Low traffic
- 5-20 Mbps - Moderate traffic
- 20-40 Mbps - High traffic
- > 40 Mbps - Peak traffic

Grafana Query Example:

```
from(bucket: "nokia-monitor")
  |> range(start: -1h)
  |> filter(fn: (r) => r["metricCounter"] == "M8012C26")
  |> filter(fn: (r) => r._field == "counterValue")
  |> map(fn: (r) => ({ r with _value: r._value / 1000.0}))  //
Convert to Mbps
```

UE Activity Counters

M8018 - eNB Load Measurements

These counters track the number of active User Equipment (UE) devices connected to the base station.

Counter	Name	Description	Unit	Type	Update Interval
M8018C1	Active UE per eNB max	Maximum number of active UEs per eNodeB	Integer	Max	1 second

Understanding UE Activity:

- **Active UE** = A device with at least one Signaling Radio Bearer (SRB) and one Data Radio Bearer (DRB)
- Maximum value observed over 1-second sampling periods
- Indicates peak concurrent user load

Load Levels:

Active UEs	Load Level	Recommendation
0-50	Low	Normal operation
50-100	Moderate	Monitor capacity
100-150	High	Evaluate adding capacity
> 150	Very High	Capacity expansion needed

Notes:

- Actual capacity depends on base station hardware configuration
- Nokia AirScale can typically support 150-250 concurrent active UEs per cell
- High UE counts may impact per-user throughput

Cell Availability Counters

M8020 - Cell Availability Measurements

These counters calculate cell availability percentage by sampling cell state at regular intervals.

Counter	Name	Description	Unit	Type	Updat Interv
M8020C3	Samples when the cell is available	Count of samples when cell was available	Integer	Sum	~10 second
M8020C4	Samples when the cell is planned unavailable	Count of samples when cell was in planned maintenance	Integer	Sum	~10 second
M8020C6	Cell availability denominator	Total number of availability check samples	Integer	Denominator	~10 second

Calculating Cell Availability:

```
Cell Availability % = 100.0 × M8020C3 / (M8020C6 - M8020C4)
```

Formula Explanation:

- **M8020C3** - Samples when cell was operating normally
- **M8020C6** - Total samples taken
- **M8020C4** - Planned downtime samples (excluded from calculation)

Availability Targets:

Availability	Grade	Status
> 99.9%	Excellent	Meeting carrier-grade SLA
99.0-99.9%	Good	Normal operations
95.0-99.0%	Fair	Investigate issues
< 95.0%	Poor	Critical - immediate action required

Grafana Query Example:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] == "M8020C3" or
                       r["metricCounter"] == "M8020C6" or
                       r["metricCounter"] == "M8020C4")
  |> pivot(rowKey:["_time"], columnKey: ["metricCounter"],
valueColumn: "_value")
  |> map(fn: (r) => ({
      _time: r._time,
      "Cell Availability": 100.0 * r.M8020C3 / (r.M8020C6 -
r.M8020C4)
    }))
```

Radio Quality Counters

M8005 - Radio Quality Measurements

These counters measure Received Signal Strength Indicator (RSSI) and Signal to Interference and Noise Ratio (SINR), providing insight into radio conditions.

RSSI Measurements

Counter	Name	Description	Unit	Type
M8005C0	RSSI for PUCCH Min	Minimum RSSI on Physical Uplink Control Channel	dBm	Min
M8005C1	RSSI for PUCCH Max	Maximum RSSI on Physical Uplink Control Channel	dBm	Max
M8005C2	RSSI for PUCCH Mean	Mean RSSI on Physical Uplink Control Channel	dBm	Average

Understanding RSSI:

- **RSSI** = Received Signal Strength Indicator (total received power)
- **PUCCH** = Physical Uplink Control Channel (carries control information)
- Measured in dBm (decibel-milliwatts)
- Updated when UE-related RSSI values are calculated

RSSI Value Interpretation:

RSSI Range	Quality	Description
> -70 dBm	Excellent	Very strong signal
-70 to -85 dBm	Good	Strong signal, good performance
-85 to -100 dBm	Fair	Adequate signal
-100 to -110 dBm	Poor	Weak signal, potential issues
< -110 dBm	Very Poor	Very weak signal, likely problems

Use Cases:

- **Coverage analysis** - Low RSSI indicates coverage gaps
- **Interference troubleshooting** - Unexpected RSSI patterns

- **RF planning** - Validate signal strength predictions

Connection Management Counters

M8013 - Signaling Connection Establishment

These counters track RRC (Radio Resource Control) connection setup attempts and successes, key indicators of network accessibility.

Counter	Name	Description	Unit	Type
M8013C5	Signaling Connection Establishment Completions	Successful RRC connection setups	Integer	Sum
M8013C17	Signaling Connection Establishment Attempts MO-S	Connection attempts - Mobile Originated Signaling	Integer	Sum
M8013C18	Signaling Connection Establishment Attempts MT	Connection attempts - Mobile Terminated	Integer	Sum
M8013C19	Signaling Connection Establishment Attempts MO-D	Connection attempts - Mobile Originated Data	Integer	Sum
M8013C21	Signaling Connection Establishment Attempts Emergency	Emergency call connection attempts	Integer	Sum
M8013C31	Signaling Connection Establishment Attempts High Priority	High priority connection attempts	Integer	Sum
M8013C34	Signaling Connection Establishment Attempts Delay Tolerant	Delay tolerant connection attempts	Integer	Sum
M8013C91	Signaling Connection Establishment Attempts MO-V	Connection attempts - Mobile Originated Voice	Integer	Sum

Counter	Name	Description	Unit	Type
M8013C93	Signaling Connection Establishment Attempts MT-Access	Connection attempts - MT Access	Integer	Sum

Calculating Setup Success Rate:

```
Setup Success Ratio % = 100.0 × M8013C5 / (M8013C17 + M8013C18 + M8013C19 + M8013C34 + M8013C31 + M8013C21 + M8013C93 + M8013C91)
```

Formula Explanation:

- **M8013C5** - Successful completions (RRC Connection Setup Complete received)
- **Sum of attempt counters** - Total connection attempts across all categories

Connection Types:

- **MO-S (M8013C17)** - Mobile Originated Signaling (SMS, location updates)
- **MT (M8013C18)** - Mobile Terminated (incoming calls/data)
- **MO-D (M8013C19)** - Mobile Originated Data (data sessions)
- **MO-V (M8013C91)** - Mobile Originated Voice (VoLTE calls)
- **Emergency (M8013C21)** - Emergency calls (911, 112)

Performance Targets:

Success Rate	Grade	Status
> 99%	Excellent	Normal operation
95-99%	Good	Acceptable performance
90-95%	Fair	Investigation recommended
< 90%	Poor	Critical issue - troubleshoot immediately

Common Failure Causes:

- Coverage issues (weak signal)
- Congestion (cell at capacity)
- Configuration errors
- Hardware problems
- Interference

Latency and QoS Counters

M8001 - Cell Performance Measurements

Counter	Name	Description	Unit	Type
M8001C2	PDCP SDU delay on DL DTCH Mean	Mean downlink PDCP SDU retention time in eNB	ms	Average

Understanding Latency:

- **PDCP SDU** = Packet Data Convergence Protocol Service Data Unit (user data packet)
- **Delay** = Time packet spends in base station before transmission
- **DL DTCH** = Downlink Dedicated Traffic Channel (user data channel)

- Lower values = better responsiveness

Latency Targets:

Latency	Grade	Application Impact
< 10 ms	Excellent	Ideal for VoLTE, gaming, video calls
10-20 ms	Good	Acceptable for most applications
20-50 ms	Fair	Noticeable in interactive apps
> 50 ms	Poor	Impacts real-time applications

Causes of High Latency:

- Queue congestion (too many users)
- Scheduler configuration issues
- Poor radio conditions (many retransmissions)
- Backhaul delays

S1 Interface Counters

M8000 - S1 Interface Measurements

These counters track the S1 interface between the eNodeB and the MME (Mobility Management Entity), including context setup, S1 connection management, and signaling procedures.

Counter	Name	Description	Unit	Type
M8000C0	Initial Context Setup Requests	Number of Initial Context Setup attempts	Integer	Sum
M8000C1	Initial Context Setup Completions	Successful Initial Context Setups	Integer	Sum
M8000C2	Setup Failures - Radio Network	Failures due to radio network issues	Integer	Sum
M8000C3	Setup Failures - Transport	Failures due to transport layer issues	Integer	Sum
M8000C6	S1 Setup Requests	S1 interface establishment attempts	Integer	Sum
M8000C7	S1 Setup Completions	Successful S1 setups	Integer	Sum
M8000C11	S1 Paging Requests	Paging messages from MME	Integer	Sum
M8000C12	UE Logical S1 Connection Setup	UE-associated S1 connections established	Integer	Sum
M8000C29	Uplink NAS Transport	NAS messages sent to MME	Integer	Sum
M8000C30	Downlink NAS Transport	NAS messages received from MME	Integer	Sum

Understanding S1 Interface:

- **S1** interface connects eNodeB to the EPC (Evolved Packet Core)
- **Initial Context Setup** establishes the context for a new UE connection
- **S1 Setup** is the initial handshake between eNodeB and MME
- **NAS (Non-Access Stratum)** messages carry higher-layer signaling

Success Rate Calculation:

```
S1 Setup Success Rate = 100 × M8000C7 / M8000C6
Initial Context Success Rate = 100 × M8000C1 / M8000C0
```

Performance Targets:

- S1 Setup Success Rate: > 99%
- Initial Context Success Rate: > 95%

EPS Bearer Counters

M8006 - EPS Bearer Measurements

These counters track E-UTRAN Radio Access Bearer (E-RAB) and EPS bearer establishment, modification, and release procedures.

Key Counters:

Counter	Name	Description	Unit	Type
M8006C0	EPS Bearer Setup Attempts	Bearer establishment attempts	Integer	Sum
M8006C1	EPS Bearer Setup Completions	Successful bearer setups	Integer	Sum
M8006C2-C5	Setup Failures by Cause	Failures categorized by reason	Integer	Sum

Understanding EPS Bearers:

- **Bearer** = Logical channel for user data between UE and network
- **Default Bearer** = Always-on bearer for internet connectivity
- **Dedicated Bearer** = Additional bearers for specific QoS requirements (VoLTE, video streaming)

Use Cases:

- Monitor bearer setup success rates
- Identify reasons for bearer failures
- Track QoS bearer allocation for premium services

Handover Counters

M8009 - Handover Preparation Measurements

M8014 - Inter-eNB Handover Measurements

M8015 - Intra-eNB Handover Measurements

M8021 - Inter-Frequency Handover Measurements

These measurement groups track handover procedures - the process of transferring a UE connection from one cell to another without dropping the call.

Handover Types:

Type	Description	Measurement Group
Intra-eNB	Handover between cells on same base station	M8015
Inter-eNB	Handover between different base stations (X2)	M8014
Inter-Frequency	Handover to different carrier frequency	M8021
Inter-RAT	Handover to different technology (LTE→3G/2G)	M8017

Key Metrics:

Counter Group	Focus	Critical Counters
M8009	Handover prep failures	Prep attempts, failures by cause
M8014	X2-based HO	Prep attempts, executions, failures
M8015	Intra-cell HO	Prep attempts, executions, failures
M8021	Inter-freq HO	Attempts, successes, failures

Handover Success Rate Formula:

```
HO Success Rate = 100 × (HO Executions) / (HO Prep Attempts)
```

Performance Targets:

- Intra-eNB HO Success Rate: > 99%
- Inter-eNB HO Success Rate: > 98%
- Inter-Frequency HO Success Rate: > 95%

Common Handover Failure Causes:

- Target cell congestion (no resources available)
- Poor radio conditions at target cell
- Timer expiration (UE doesn't respond in time)
- Measurement gaps preventing proper cell selection

Channel Quality Measurements

M8010 - CQI (Channel Quality Indicator) Distribution

These counters track the distribution of CQI reports from UEs, providing insight into radio channel quality.

Understanding CQI:

- **CQI** = Channel Quality Indicator reported by UE to eNodeB
- **Range:** CQI 0 (worst) to CQI 15 (best)
- **Purpose:** Helps scheduler select appropriate MCS (Modulation and Coding Scheme)
- **Update Frequency:** Every few milliseconds based on channel conditions

CQI to Data Rate Mapping:

CQI Level	Channel Quality	Approximate Max Rate	Modulation
0-3	Very Poor	< 1 Mbps	QPSK
4-6	Poor	1-5 Mbps	QPSK
7-9	Fair	5-15 Mbps	16-QAM
10-12	Good	15-40 Mbps	64-QAM
13-15	Excellent	40-150 Mbps	64-QAM

M8010 Counters:

- M8010C0 - M8010C15: Count of CQI reports at each level (0-15)

Performance Analysis:

```
Average CQI = Σ(CQI_level × M8010C[level]) / Σ(M8010C[level])
```

Interpreting CQI Distribution:

- **High CQI (10-15):** Good coverage, high throughput potential
- **Medium CQI (7-9):** Adequate coverage, moderate throughput
- **Low CQI (0-6):** Coverage issues, consider cell optimization

---

CS Fallback Counters

M8016 - CS Fallback Measurements

These counters track Circuit-Switched Fallback (CSFB) procedures, where LTE UEs fall back to 2G/3G networks for voice calls.

Understanding CSFB:

- **Purpose:** Handle voice calls on networks without VoLTE
- **Process:** UE in LTE → Voice call → Temporarily move to 2G/3G → Return to LTE
- **Impact:** Call setup delay, temporary loss of LTE data

Key Counters:

Counter	Name	Description
M8016C0	CSFB Attempts with Redirection	CSFB using redirection method
M8016C1	CSFB Attempts with Handover	CSFB using handover method

CSFB Methods:

1. **Redirection:** UE disconnects from LTE, re-selects 2G/3G (faster but brief service interruption)
2. **Handover:** Proper handover from LTE to 2G/3G (slower but seamless)

Performance Metrics:

- CSFB Success Rate = Successful transitions / Total CSFB attempts
- Target: > 98%

Data Volume Counters

M8023 - PDCP SDU Data Volume Measurements

These counters track the total volume of user data transmitted over the air interface.

Key Metrics:

- Total data volume (uplink and downlink)
- Volume per QCI (Quality of Service Class Identifier)
- Volume per bearer type

Use Cases:

- Network capacity planning
- Revenue estimation (data usage tracking)
- Per-user data consumption analysis
- QoS class traffic profiling

Relationship to M8012 (Throughput):

- **M8012:** Instantaneous data rate (kbit/s)
- **M8023:** Cumulative data volume (bytes)

X2 Interface Counters

M8004 - X2 Data Volume Measurements

M8022 - X2 Setup Measurements

These counters track the X2 interface between eNodeBs, used for inter-eNB handovers and load balancing.

M8004 - X2 Data Volume:

- Measures data forwarded between eNodeBs during handovers
- Tracks incoming and outgoing X2 traffic volume

M8022 - X2 Setup:

- **M8022C0:** X2 Setup Attempts
- **M8022C1:** X2 Setup Successes

Understanding X2 Interface:

- **Purpose:** Direct communication between neighboring eNodeBs
- **Functions:** Handover coordination, load sharing, interference management
- **Benefit:** Reduces core network load, faster handovers

X2 Setup Success Rate:

```
X2 Setup Success Rate = 100 × M8022C1 / M8022C0
```

Target: > 95%

Additional Measurement Groups

M8007 - Data Radio Bearer (DRB) Measurements

Tracks the establishment, modification, and release of Data Radio Bearers (DRBs) for user data transmission.

Focus Areas:

- DRB setup success rates
- DRB modification procedures
- Bearer release statistics

M8008 - RRC Connection Rejection Measurements

Tracks RRC connection requests that are rejected, categorized by rejection cause.

Common Rejection Reasons:

- Network congestion (insufficient PRBs)
- Maximum UE limit reached
- Load balancing redirection
- Mobility restrictions

M8019 - Network Assisted Cell Change (NACC)

Tracks NACC procedures for optimizing cell reselection from LTE to GSM.

Purpose: Provides GSM neighbor cell system information to UEs before reselection, speeding up the transition.

Network Interface Measurements

M5112 - IP Interface Input Measurements

M5113 - Ethernet RX Measurements

These measurement groups track backhaul interface statistics.

M5112 - IP Interface Input (112 counters):

- Incoming packet counts
- Packet size distribution
- Protocol-specific statistics
- Interface utilization

M5113 - Ethernet RX (21 counters):

- Received Ethernet frame counts
- Frame size distribution
- Error statistics (CRC errors, frame errors)

Use Cases:

- Backhaul capacity monitoring
- Transport link health assessment
- Troubleshooting connectivity issues
- Capacity planning for transport upgrades

Hardware and Radio Unit Counters

M40001 - Radio Hardware Measurements

Counter	Name	Description	Unit	Type	Scaling
M40001C0	VSWR per antenna branch	Voltage Standing Wave Ratio	0.1	Average	Divide by 10

Understanding VSWR:

- **VSWR** = Voltage Standing Wave Ratio
- Measures antenna system efficiency
- Indicates impedance mismatch and potential cable/antenna problems
- Lower values = better

VSWR Interpretation:

VSWR	Status	Action
1.0-1.5	Excellent	Normal operation
1.5-2.0	Good	Acceptable
2.0-3.0	Fair	Investigate
> 3.0	Poor	Cable/antenna issue - troubleshoot immediately

Common Causes of High VSWR:

- Damaged coaxial cables
- Loose connectors
- Water ingress
- Antenna damage

- Impedance mismatch

Grafana Query Example:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] == "M40001C0")
  |> filter(fn: (r) => r._field == "counterValue")
  |> map(fn: (r) => ({ r with "VSWR": r._value / 10.0}))
```

M40002 - Power Consumption Measurements

Counter	Name	Description	Unit	Type	Scaling
M40002C2	Power Consumption	Base station power consumption	100000 factor	Average	Divide by 100000

Understanding Power Consumption:

- Measures total base station power draw
- Useful for OPEX calculations and capacity planning
- Can indicate hardware problems if unexpectedly high/low

Grafana Query Example:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] == "M40002C2")
  |> filter(fn: (r) => r._field == "counterValue")
  |> map(fn: (r) => ({ r with "Power": r._value / 100000.0}))
```

Using Counters in Grafana

Building Effective Dashboards

1. Resource Utilization Dashboard

Combine M8011C24 and M8011C37 to show uplink/downlink PRB usage:

```
// Uplink PRB Usage
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] == "M8011C24")
  |> map(fn: (r) => ({ r with _value: r._value / 10.0}))
  |> rename(columns: {"_value": "Uplink PRB %"})

// Downlink PRB Usage
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] == "M8011C37")
  |> map(fn: (r) => ({ r with _value: r._value / 10.0}))
  |> rename(columns: {"_value": "Downlink PRB %"})
```

2. Throughput Dashboard

Show data transfer rates:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] == "M8012C23" or
r["metricCounter"] == "M8012C26")
  |> map(fn: (r) => ({ r with _value: r._value / 1000.0}))  //
Convert to Mbps
  |> pivot(rowKey: ["_time"], columnKey: ["metricCounter"],
valueColumn: "_value")
  |> map(fn: (r) => ({
      _time: r._time,
      "Uplink Mbps": r.M8012C23,
      "Downlink Mbps": r.M8012C26
    }))
```

3. Availability Dashboard

Calculate and display cell availability:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] =~ /M8020C(3|4|6)/)
  |> pivot(rowKey: ["_time"], columnKey: ["metricCounter"],
valueColumn: "_value")
  |> map(fn: (r) => ({
      _time: r._time,
      "Availability %": 100.0 * r.M8020C3 / (r.M8020C6 -
r.M8020C4)
    }))
```

4. Connection Success Rate Dashboard

Track RRC setup performance:

```
from(bucket: "nokia-monitor")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["metricCounter"] =~
/M8013C(5|17|18|19|21|31|34|91|93)/)
  |> pivot(rowKey: ["_time"], columnKey: ["metricCounter"],
valueColumn: "_value")
  |> map(fn: (r) => ({
      _time: r._time,
      "Setup Success %": 100.0 * r.M8013C5 / (r.M8013C17 +
r.M8013C18 + r.M8013C19 + r.M8013C34 + r.M8013C31 + r.M8013C21 +
r.M8013C93 + r.M8013C91)
    }))
```

Dashboard Best Practices

Use Appropriate Visualization Types:

- Time series graphs - Trend data (throughput, PRB usage)
- Gauges - Current values (availability percentage)
- Single stat panels - Peak values (max active UEs)

- Heat maps - Distribution data (RSSI levels)

Set Meaningful Thresholds:

- Green: Normal operation
- Yellow: Warning (investigate)
- Red: Critical (immediate action)

Example Threshold Configuration:

- PRB utilization: Green < 70%, Yellow 70-85%, Red > 85%
- Availability: Green > 99%, Yellow 95-99%, Red < 95%
- Setup success: Green > 99%, Yellow 95-99%, Red < 95%

Group Related Metrics:

- Create separate dashboards for capacity, performance, and quality
- Use template variables for site/cell selection
- Include drill-down links to detailed views

Related Documentation

- Operations Guide - RAN Monitor operations and workflows
- Grafana Integration Guide - Setting up Grafana dashboards
- Runtime Configuration Guide - Configuring RAN Monitor
- AirScale Configuration Guide - Base station setup
- Metrics Reference - InfluxDB data structure

Quick Reference Table

Most Commonly Used Counters

Counter	Name	Use Case	Unit
M8011C24	UL PRB util mean	Capacity planning	0.1%
M8011C37	DL PRB util mean	Capacity planning	0.1%
M8012C23	PDCP UL throughput	Performance monitoring	kbit/s
M8012C26	PDCP DL throughput	Performance monitoring	kbit/s
M8018C1	Active UEs max	Load monitoring	count
M8020C3	Cell available samples	Availability tracking	count
M8020C6	Availability denominator	Availability calculation	count
M8013C5	Setup completions	Success rate tracking	count
M8005C0	RSSI min	Coverage analysis	dBm
M8005C2	RSSI mean	Coverage analysis	dBm
M8001C2	PDCP delay mean	Latency monitoring	ms
M40001C0	VSWR	Hardware health	0.1 ratio

Sources:

- Nokia FlexiRadio LTE Performance Measurements
- Nokia AirScale Performance Counter Specifications
- 3GPP LTE Performance Management Standards

Appendix: Complete Counter Listing

Below is the complete reference listing of all 1,186 Nokia LTE performance counters extracted from the Nokia KPI specification.

```
=========================================================================
NOKIA LTE PERFORMANCE COUNTER COMPLETE REFERENCE
=========================================================================

-------------------------------------------------------------------------
Measurement Group: M8000
-------------------------------------------------------------------------
M8000C0         | Initial Context Setup requests
M8000C1         | Initial Context Setup completions
M8000C2         | Initial Context Setup failures due to
M8000C3         | Initial Context Setup failures due to
M8000C4         | Initial Context Setup failures due to
M8000C5         | Initial Context Setup failures due to
M8000C6         | S1 setup requests
M8000C7         | S1 setup completions
M8000C8         | S1 setup failure due to timer expiration
M8000C9         | S1 setup failure due to MME rejection
M8000C11        | S1 Paging requests
M8000C12        | Number of UE-associated logical S1-
M8000C13        | Global S1 Reset initiated by the eNB
M8000C14        | Global S1 Reset initiated by the MME
M8000C15        | Partial S1 Reset initiated by the eNB
M8000C16        | Partial S1 Reset initiated by the MME
M8000C23        | UE Context modification attempts
M8000C24        | UE Context modification attempts due
M8000C25        | UE Context modification failures
M8000C29        | Number of Uplink NAS Transport
M8000C30        | Number of Downlink NAS Transport
M8000C31        | UE Context modification attempts due
M8000C32        | E-RAB Setup Requests for IMS
M8000C33        | E-RAB Setup Completions for IMS
M8000C34        | E-RAB setup failures for IMS
M8000C35        | Number of Location Reporting Control
M8000C36        | Number of Location Report messages
M8000C37        | Number of successful X2 IP address
M8000C38        | Number of attempted X2 IP address
M8000C39        | Number of WRITE-REPLACE
M8000C40        | Number of WRITE-REPLACE
M8000C41        | Number of KILL REQUEST messages
M8000C42        | Number of KILL RESPONSE

-------------------------------------------------------------------------
Measurement Group: M8001
```

```
-----------------------------------------------------------------------
M8001C0         | PDCP SDU delay on DL DTCH Min
M8001C1         | PDCP SDU delay on DL DTCH Max
M8001C2         | PDCP SDU delay on DL DTCH Mean
M8001C3         | PDCP SDU delay on UL DTCH Min
M8001C4         | PDCP SDU delay on UL DTCH Max
M8001C5         | PDCP SDU delay on UL DTCH Mean
M8001C6         | RACH setup attempts for small size
M8001C7         | RACH setup attempts for large size
M8001C8         | RACH setup completions
M8001C9         | Transmitted TBs on PCH
M8001C10        | Transmitted TBs on BCH
M8001C11        | Transmitted TBs on DL-SCH
M8001C12        | HARQ retransmissions on DL-SCH
M8001C13        | Correct non-duplicate UL-SCH TB with
M8001C14        | Correct UL-SCH TB with re-reception
M8001C15        | Erroneous UL-SCH TB receptions
M8001C16        | PUSCH transmissions using MCS0
M8001C17        | PUSCH transmissions using MCS1
M8001C18        | PUSCH transmissions using MCS2
M8001C19        | PUSCH transmissions using MCS3
M8001C20        | PUSCH transmissions using MCS4
M8001C21        | PUSCH transmissions using MCS5
M8001C22        | PUSCH transmissions using MCS6
M8001C23        | PUSCH transmissions using MCS7
M8001C24        | PUSCH transmissions using MCS8
M8001C25        | PUSCH transmissions using MCS9
M8001C26        | PUSCH transmissions using MCS10
M8001C27        | PUSCH transmissions using MCS11
M8001C28        | PUSCH transmissions using MCS12
M8001C29        | PUSCH transmissions using MCS13
M8001C30        | PUSCH transmissions using MCS14
M8001C31        | PUSCH transmissions using MCS15
M8001C32        | PUSCH transmissions using MCS16
M8001C33        | PUSCH transmissions using MCS17
M8001C34        | PUSCH transmissions using MCS18
M8001C35        | PUSCH transmissions using MCS19
M8001C36        | PUSCH transmissions using MCS20
M8001C37        | PUSCH transmissions using MCS21
M8001C38        | PUSCH transmissions using MCS22
M8001C39        | PUSCH transmissions using MCS23
M8001C40        | PUSCH transmissions using MCS24
M8001C41        | PUSCH transmissions using MCS25
M8001C42        | PUSCH transmissions using MCS26
```

```
M8001C43        | PUSCH transmissions using MCS27
M8001C44        | PUSCH transmissions using MCS28
M8001C45        | PDSCH transmissions using MCS0
M8001C46        | PDSCH transmissions using MCS1
M8001C47        | PDSCH transmissions using MCS2
M8001C48        | PDSCH transmissions using MCS3
M8001C49        | PDSCH transmissions using MCS4
M8001C50        | PDSCH transmissions using MCS5
M8001C51        | PDSCH transmissions using MCS6
M8001C52        | PDSCH transmissions using MCS7
M8001C53        | PDSCH transmissions using MCS8
M8001C54        | PDSCH transmissions using MCS9
M8001C55        | PDSCH transmissions using MCS10
M8001C56        | PDSCH transmissions using MCS11
M8001C57        | PDSCH transmissions using MCS12
M8001C58        | PDSCH transmissions using MCS13
M8001C59        | PDSCH transmissions using MCS14
M8001C60        | PDSCH transmissions using MCS15
M8001C61        | PDSCH transmissions using MCS16
M8001C62        | PDSCH transmissions using MCS17
M8001C63        | PDSCH transmissions using MCS18
M8001C64        | PDSCH transmissions using MCS19
M8001C65        | PDSCH transmissions using MCS20
M8001C66        | PDSCH transmissions using MCS21
M8001C67        | PDSCH transmissions using MCS22
M8001C68        | PDSCH transmissions using MCS23
M8001C69        | PDSCH transmissions using MCS24
M8001C70        | PDSCH transmissions using MCS25
M8001C71        | PDSCH transmissions using MCS26
M8001C72        | PDSCH transmissions using MCS27
M8001C73        | PDSCH transmissions using MCS28
M8001C74        | PUSCH transmission nacks using
M8001C75        | PUSCH transmission nacks using
M8001C76        | PUSCH transmission nacks using
M8001C77        | PUSCH transmission nacks using
M8001C78        | PUSCH transmission nacks using
M8001C79        | PUSCH transmission nacks using
M8001C80        | PUSCH transmission nacks using
M8001C81        | PUSCH transmission nacks using
M8001C82        | PUSCH transmission nacks using
M8001C83        | PUSCH transmission nacks using
M8001C84        | PUSCH transmission nacks using
M8001C85        | PUSCH transmission nacks using
M8001C86        | PUSCH transmission nacks using
```

M8001C87	PUSCH transmission nacks using
M8001C88	PUSCH transmission nacks using
M8001C89	PUSCH transmission nacks using
M8001C90	PUSCH transmission nacks using
M8001C91	PUSCH transmission nacks using
M8001C92	PUSCH transmission nacks using
M8001C93	PUSCH transmission nacks using
M8001C94	PUSCH transmission nacks using
M8001C95	PUSCH transmission nacks using
M8001C96	PUSCH transmission nacks using
M8001C97	PUSCH transmission nacks using
M8001C98	PUSCH transmission nacks using
M8001C99	PUSCH transmission nacks using
M8001C100	PUSCH transmission nacks using
M8001C101	PUSCH transmission nacks using
M8001C102	PUSCH transmission nacks using
M8001C103	PDSCH transmission nacks using
M8001C104	PDSCH transmission nacks using
M8001C105	PDSCH transmission nacks using
M8001C106	PDSCH transmission nacks using
M8001C107	PDSCH transmission nacks using
M8001C108	PDSCH transmission nacks using
M8001C109	PDSCH transmission nacks using
M8001C110	PDSCH transmission nacks using
M8001C111	PDSCH transmission nacks using
M8001C112	PDSCH transmission nacks using
M8001C113	PDSCH transmission nacks using
M8001C114	PDSCH transmission nacks using
M8001C115	PDSCH transmission nacks using
M8001C116	PDSCH transmission nacks using
M8001C117	PDSCH transmission nacks using
M8001C118	PDSCH transmission nacks using
M8001C119	PDSCH transmission nacks using
M8001C120	PDSCH transmission nacks using
M8001C121	PDSCH transmission nacks using
M8001C122	PDSCH transmission nacks using
M8001C123	PDSCH transmission nacks using
M8001C124	PDSCH transmission nacks using
M8001C125	PDSCH transmission nacks using
M8001C126	PDSCH transmission nacks using
M8001C127	PDSCH transmission nacks using
M8001C128	PDSCH transmission nacks using
M8001C129	PDSCH transmission nacks using
M8001C130	PDSCH transmission nacks using

```
M8001C131       | PDSCH transmission nacks using
M8001C132       | RLC SDUs on DL DTCH
M8001C133       | RLC SDUs on DL DCCH
M8001C135       | RLC SDUs on UL DTCH
M8001C136       | RLC SDUs on UL DCCH
M8001C137       | RLC PDU first transmissions
M8001C138       | RLC PDU re-transmissions
M8001C139       | RLC PDU reception
M8001C140       | DL RLC C-PDUs transmitted the first
M8001C141       | DL RLC Data PDUs transmitted first
M8001C142       | UL RLC PDUs received
M8001C143       | Duplicate UL RLC PDU received
M8001C144       | UL RLC PDU retransmission requested
M8001C145       | UL RLC PDUs discarded
M8001C146       | DL RLC SDU From PDCP
M8001C147       | Average number of UEs with buffered
M8001C148       | Max number of UEs with buffered data
M8001C150       | Average number of UEs with buffered
M8001C151       | Max number of UEs with buffered data
M8001C153       | PDCP SDUs in Uplink
M8001C154       | PDCP SDUs in Downlink
M8001C155       | Discarded PDCP SDUs in downlink
M8001C156       | Failed Transmission PDSCH MCS0
M8001C157       | Failed Transmission PDSCH MCS1
M8001C158       | Failed Transmission PDSCH MCS2
M8001C159       | Failed Transmission PDSCH MCS3
M8001C160       | Failed Transmission PDSCH MCS4
M8001C161       | Failed Transmission PDSCH MCS5
M8001C162       | Failed Transmission PDSCH MCS6
M8001C163       | Failed Transmission PDSCH MCS7
M8001C164       | Failed Transmission PDSCH MCS8
M8001C165       | Failed Transmission PDSCH MCS9
M8001C166       | Failed Transmission PDSCH MCS10
M8001C167       | Failed Transmission PDSCH MCS11
M8001C168       | Failed Transmission PDSCH MCS12
M8001C169       | Failed Transmission PDSCH MCS13
M8001C170       | Failed Transmission PDSCH MCS14
M8001C171       | Failed Transmission PDSCH MCS15
M8001C172       | Failed Transmission PDSCH MCS16
M8001C173       | Failed Transmission PDSCH MCS17
M8001C174       | Failed Transmission PDSCH MCS18
M8001C175       | Failed Transmission PDSCH MCS19
M8001C176       | Failed Transmission PDSCH MCS20
M8001C177       | Failed Reception PUSCH MCS0
```

```
M8001C178       | Failed Reception PUSCH MCS1
M8001C179       | Failed Reception PUSCH MCS2
M8001C180       | Failed Reception PUSCH MCS3
M8001C181       | Failed Reception PUSCH MCS4
M8001C182       | Failed Reception PUSCH MCS5
M8001C183       | Failed Reception PUSCH MCS6
M8001C184       | Failed Reception PUSCH MCS7
M8001C185       | Failed Reception PUSCH MCS8
M8001C186       | Failed Reception PUSCH MCS9
M8001C187       | Failed Reception PUSCH MCS10
M8001C188       | Failed Reception PUSCH MCS11
M8001C189       | Failed Reception PUSCH MCS12
M8001C190       | Failed Reception PUSCH MCS13
M8001C191       | Failed Reception PUSCH MCS14
M8001C192       | Failed Reception PUSCH MCS15
M8001C193       | Failed Reception PUSCH MCS16
M8001C194       | Failed Reception PUSCH MCS17
M8001C195       | Failed Reception PUSCH MCS18
M8001C196       | Failed Reception PUSCH MCS19
M8001C197       | Failed Reception PUSCH MCS20
M8001C199       | RRC Connected UEs Avg
M8001C200       | RRC Connected UEs Max
M8001C202       | Failed Transmission PDSCH MCS21
M8001C203       | Failed Transmission PDSCH MCS22
M8001C204       | Failed Transmission PDSCH MCS23
M8001C205       | Failed Transmission PDSCH MCS24
M8001C206       | Failed Transmission PDSCH MCS25
M8001C207       | Failed Transmission PDSCH MCS26
M8001C208       | Failed Transmission PDSCH MCS27
M8001C209       | Failed Transmission PDSCH MCS28
M8001C210       | Minimum queue length DL SRB
M8001C211       | Average queue length DL SRB
M8001C212       | Maximum queue length DL SRB
M8001C213       | Minimum queue length DL DRB
M8001C214       | Average queue length DL DRB
M8001C215       | Maximum queue length DL DRB
M8001C216       | Average number of available PRBs per
M8001C217       | Average number of available PRBs per
M8001C218       | RLC SDUs on BCCH
M8001C219       | RLC SDUs on DL CCCH
M8001C220       | RLC SDUs on PCCH
M8001C222       | RLC SDUs on UL CCCH
M8001C223       | Active UE per Cell average
M8001C224       | Active UE per Cell max
```

```
M8001C227       | UEs with buffered DL data for DRB with
M8001C228       | UEs with buffered DL data for QCI2
M8001C229       | UEs with buffered DL data for QCI3
M8001C230       | UEs with buffered DL data for QCI4
M8001C231       | Number of primary ETWS notifications
M8001C232       | Number of secondary ETWS
M8001C233       | Number of CMAS notifications
M8001C235       | UEs with buffered DL data for non-GBR
M8001C254       | Number of lost PDCP SDUs in UL
M8001C255       | Number of lost PDCP SDUs in UL for
M8001C256       | Number of lost PDCP SDUs in UL for
M8001C257       | Number of lost PDCP SDUs in UL for
M8001C258       | Number of lost PDCP SDUs in UL for
M8001C259       | Number of lost PDCP SDUs in DL
M8001C260       | Number of lost PDCP SDUs in DL for
M8001C261       | Number of lost PDCP SDUs in DL for
M8001C262       | Number of lost PDCP SDUs in DL for
M8001C263       | Number of lost PDCP SDUs in DL for
M8001C264       | Sum of Active UEs with buffered data in
M8001C265       | Denominator for Active UEs with
M8001C266       | Sum of Active UEs with buffered data in
M8001C267       | Denominator for Active UEs with
M8001C269       | Mean PDCP SDU delay on DL DTCH
M8001C270       | Mean PDCP SDU delay on DL DTCH
M8001C271       | PDCP SDU delay on DL DTCH Mean
M8001C272       | PDCP SDU delay on DL DTCH Mean
M8001C273       | PDCP SDU delay on DL DTCH Mean
M8001C286       | RACH setup attempts for dedicated
M8001C305       | PDCP SDU UL QCI 1
M8001C306       | PDCP SDU UL QCI 2
M8001C307       | PDCP SDU UL QCI 3
M8001C308       | PDCP SDU UL QCI 4
M8001C314       | PDCP SDU DL QCI 1
M8001C315       | PDCP SDU DL QCI 2
M8001C316       | PDCP SDU DL QCI 3
M8001C317       | PDCP SDU DL QCI 4
M8001C318       | Sum of RRC Connected UEs per cell
M8001C319       | Denominator for RRC Connected UEs
M8001C320       | Sum of Active UEs per cell
M8001C321       | Denominator for Active UEs per cell
M8001C323       | PDCP SDU discarded DL QCI 1
M8001C324       | PDCP SDU discarded DL QCI 2
M8001C325       | PDCP SDU discarded DL QCI 3
M8001C326       | PDCP SDU discarded DL QCI 4
```

```
M8001C419       | UEs with buffered UL data for DRB with
M8001C420       | UEs with buffered UL data for non-GBR
M8001C421       | Number of PDCCH order attempts
M8001C422       | Number of initial PDCCH order
M8001C423       | Number of successful PDCCH orders
M8001C424       | Unavailability of dedicated preamble
M8001C425       | Unavailability of dedicated preamble for
M8001C426       | Unavailability of dedicated preamble for
M8001C427       | Number of UE state transitions from
M8001C428       | Average number of UEs in UL-In-Sync
M8001C429       | Average number of UEs with unlimited
M8001C430       | UE UL-Out-Of-Sync time T in the range
M8001C431       | UE UL-Out-Of-Sync time T in the range
M8001C432       | UE UL-Out-Of-Sync time T in the range
M8001C433       | UE UL-Out-Of-Sync time T in the range
M8001C434       | UE UL-Out-Of-Sync time T larger than
M8001C435       | First transmissions on PUSCH using
M8001C436       | First transmissions on PUSCH using
M8001C437       | First transmissions on PUSCH using
M8001C438       | First transmissions on PUSCH using
M8001C439       | First transmissions on PUSCH using
M8001C440       | First transmissions on PUSCH using
M8001C441       | First transmissions on PUSCH using
M8001C442       | First transmissions on PUSCH using
M8001C443       | First transmissions on PUSCH using
M8001C444       | First transmissions on PUSCH using
M8001C445       | First transmissions on PUSCH using
M8001C446       | First transmissions on PUSCH using
M8001C447       | First transmissions on PUSCH using
M8001C448       | First transmissions on PUSCH using
M8001C449       | First transmissions on PUSCH using
M8001C450       | First transmissions on PUSCH using
M8001C451       | First transmissions on PUSCH using
M8001C452       | First transmissions on PUSCH using
M8001C453       | First transmissions on PUSCH using
M8001C454       | First transmissions on PUSCH using
M8001C455       | First transmissions on PUSCH using
M8001C456       | First transmissions on PUSCH using
M8001C457       | First transmissions on PUSCH using
M8001C458       | First transmissions on PUSCH using
M8001C459       | First transmissions on PUSCH using
M8001C460       | First transmission NACKs on PUSCH
M8001C461       | First transmission NACKs on PUSCH
M8001C462       | First transmission NACKs on PUSCH
```

```
M8001C463       | First transmission NACKs on PUSCH
M8001C464       | First transmission NACKs on PUSCH
M8001C465       | First transmission NACKs on PUSCH
M8001C466       | First transmission NACKs on PUSCH
M8001C467       | First transmission NACKs on PUSCH
M8001C468       | First transmission NACKs on PUSCH
M8001C469       | First transmission NACKs on PUSCH
M8001C470       | First transmission NACKs on PUSCH
M8001C471       | First transmission NACKs on PUSCH
M8001C472       | First transmission NACKs on PUSCH
M8001C473       | First transmission NACKs on PUSCH
M8001C474       | First transmission NACKs on PUSCH
M8001C475       | First transmission NACKs on PUSCH
M8001C476       | First transmission NACKs on PUSCH
M8001C477       | First transmission NACKs on PUSCH
M8001C478       | First transmission NACKs on PUSCH
M8001C479       | First transmission NACKs on PUSCH
M8001C480       | First transmission NACKs on PUSCH
M8001C481       | First transmission NACKs on PUSCH
M8001C482       | First transmission NACKs on PUSCH
M8001C483       | First transmission NACKs on PUSCH
M8001C484       | First transmission NACKs on PUSCH
M8001C485       | Failed Reception PUSCH MCS21
M8001C486       | Failed Reception PUSCH MCS22
M8001C487       | Failed Reception PUSCH MCS23
M8001C488       | Failed Reception PUSCH MCS24
M8001C494       | Average number of DL carrier
M8001C495       | Average number of UEs with a
M8001C496       | Average number of UEs with an

-------------------------------------------------------------------------
Measurement Group: M8004
-------------------------------------------------------------------------
M8004C0         | X2 data volume per eNB, incoming
M8004C1         | X2 data volume per eNB, outgoing
M8004C2         | Incoming u-plane X2 data volume per
M8004C3         | Outgoing u-plane X2 data volume per

-------------------------------------------------------------------------
Measurement Group: M8005
-------------------------------------------------------------------------
M8005C0         | RSSI for PUCCH Min
M8005C1         | RSSI for PUCCH Max
M8005C2         | RSSI for PUCCH Mean
```

```
M8005C3         | RSSI for PUSCH Min
M8005C4         | RSSI for PUSCH Max
M8005C5         | RSSI for PUSCH Mean
M8005C6         | RSSI for PUCCH Level 01
M8005C7         | RSSI for PUCCH Level 02
M8005C8         | RSSI for PUCCH Level 03
M8005C9         | RSSI for PUCCH Level 04
M8005C10        | RSSI for PUCCH Level 05
M8005C11        | RSSI for PUCCH Level 06
M8005C12        | RSSI for PUCCH Level 07
M8005C13        | RSSI for PUCCH Level 08
M8005C14        | RSSI for PUCCH Level 09
M8005C15        | RSSI for PUCCH Level 10
M8005C16        | RSSI for PUCCH Level 11
M8005C17        | RSSI for PUCCH Level 12
M8005C18        | RSSI for PUCCH Level 13
M8005C19        | RSSI for PUCCH Level 14
M8005C20        | RSSI for PUCCH Level 15
M8005C21        | RSSI for PUCCH Level 16
M8005C22        | RSSI for PUCCH Level 17
M8005C23        | RSSI for PUCCH Level 18
M8005C24        | RSSI for PUCCH Level 19
M8005C25        | RSSI for PUCCH Level 20
M8005C26        | RSSI for PUCCH Level 21
M8005C27        | RSSI for PUCCH Level 22
M8005C28        | RSSI for PUSCH Level 01
M8005C29        | RSSI for PUSCH Level 02
M8005C30        | RSSI for PUSCH Level 03
M8005C31        | RSSI for PUSCH Level 04
M8005C32        | RSSI for PUSCH Level 05
M8005C33        | RSSI for PUSCH Level 06
M8005C34        | RSSI for PUSCH Level 07
M8005C35        | RSSI for PUSCH Level 08
M8005C36        | RSSI for PUSCH Level 09
M8005C37        | RSSI for PUSCH Level 10
M8005C38        | RSSI for PUSCH Level 11
M8005C39        | RSSI for PUSCH Level 12
M8005C40        | RSSI for PUSCH Level 13
M8005C41        | RSSI for PUSCH Level 14
M8005C42        | RSSI for PUSCH Level 15
M8005C43        | RSSI for PUSCH Level 16
M8005C44        | RSSI for PUSCH Level 17
M8005C45        | RSSI for PUSCH Level 18
M8005C46        | RSSI for PUSCH Level 19
```

```
M8005C47        | RSSI for PUSCH Level 20
M8005C48        | RSSI for PUSCH Level 21
M8005C49        | RSSI for PUSCH Level 22
M8005C54        | UE Power Headroom for PUSCH Level
M8005C55        | UE Power Headroom for PUSCH Level
M8005C56        | UE Power Headroom for PUSCH Level
M8005C57        | UE Power Headroom for PUSCH Level
M8005C58        | UE Power Headroom for PUSCH Level
M8005C59        | UE Power Headroom for PUSCH Level
M8005C60        | UE Power Headroom for PUSCH Level
M8005C61        | UE Power Headroom for PUSCH Level
M8005C62        | UE Power Headroom for PUSCH Level
M8005C63        | UE Power Headroom for PUSCH Level
M8005C64        | UE Power Headroom for PUSCH Level
M8005C65        | UE Power Headroom for PUSCH Level
M8005C66        | UE Power Headroom for PUSCH Level
M8005C67        | UE Power Headroom for PUSCH Level
M8005C68        | UE Power Headroom for PUSCH Level
M8005C69        | UE Power Headroom for PUSCH Level
M8005C70        | UE Power Headroom for PUSCH Level
M8005C71        | UE Power Headroom for PUSCH Level
M8005C72        | UE Power Headroom for PUSCH Level
M8005C73        | UE Power Headroom for PUSCH Level
M8005C74        | UE Power Headroom for PUSCH Level
M8005C75        | UE Power Headroom for PUSCH Level
M8005C76        | UE Power Headroom for PUSCH Level
M8005C77        | UE Power Headroom for PUSCH Level
M8005C78        | UE Power Headroom for PUSCH Level
M8005C79        | UE Power Headroom for PUSCH Level
M8005C80        | UE Power Headroom for PUSCH Level
M8005C81        | UE Power Headroom for PUSCH Level
M8005C82        | UE Power Headroom for PUSCH Level
M8005C83        | UE Power Headroom for PUSCH Level
M8005C84        | UE Power Headroom for PUSCH Level
M8005C85        | UE Power Headroom for PUSCH Level
M8005C87        | UE Power Headroom PUSCH Min
M8005C88        | UE Power Headroom PUSCH Max
M8005C89        | UE Power Headroom PUSCH Mean
M8005C90        | SINR for PUCCH Min
M8005C91        | SINR for PUCCH Max
M8005C92        | SINR for PUCCH Mean
M8005C93        | SINR for PUSCH Min
M8005C94        | SINR for PUSCH Max
M8005C95        | SINR for PUSCH Mean
```

```
M8005C96        | SINR for PUCCH Level 1
M8005C97        | SINR for PUCCH Level 2
M8005C98        | SINR for PUCCH Level 3
M8005C99        | SINR for PUCCH Level 4
M8005C100       | SINR for PUCCH Level 5
M8005C101       | SINR for PUCCH Level 6
M8005C102       | SINR for PUCCH Level 7
M8005C103       | SINR for PUCCH Level 8
M8005C104       | SINR for PUCCH Level 9
M8005C105       | SINR for PUCCH Level 10
M8005C106       | SINR for PUCCH Level 11
M8005C107       | SINR for PUCCH Level 12
M8005C108       | SINR for PUCCH Level 13
M8005C109       | SINR for PUCCH Level 14
M8005C110       | SINR for PUCCH Level 15
M8005C111       | SINR for PUCCH Level 16
M8005C112       | SINR for PUCCH Level 17
M8005C113       | SINR for PUCCH Level 18
M8005C114       | SINR for PUCCH Level 19
M8005C115       | SINR for PUCCH Level 20
M8005C116       | SINR for PUCCH Level 21
M8005C117       | SINR for PUCCH Level 22
M8005C118       | SINR for PUSCH Level 1
M8005C119       | SINR for PUSCH Level 2
M8005C120       | SINR for PUSCH Level 3
M8005C121       | SINR for PUSCH Level 4
M8005C122       | SINR for PUSCH Level 5
M8005C123       | SINR for PUSCH Level 6
M8005C124       | SINR for PUSCH Level 7
M8005C125       | SINR for PUSCH Level 8
M8005C126       | SINR for PUSCH Level 9
M8005C127       | SINR for PUSCH Level 10
M8005C128       | SINR for PUSCH Level 11
M8005C129       | SINR for PUSCH Level 12
M8005C130       | SINR for PUSCH Level 13
M8005C131       | SINR for PUSCH Level 14
M8005C132       | SINR for PUSCH Level 15
M8005C133       | SINR for PUSCH Level 16
M8005C134       | SINR for PUSCH Level 17
M8005C135       | SINR for PUSCH Level 18
M8005C136       | SINR for PUSCH Level 19
M8005C137       | SINR for PUSCH Level 20
M8005C138       | SINR for PUSCH Level 21
M8005C139       | SINR for PUSCH Level 22
```

```
M8005C140       | UL AMC upgrades
M8005C141       | UL AMC downgrades
M8005C206       | Minimum of cell-wide RSSI on PUCCH
M8005C207       | Maximum of cell-wide RSSI on PUCCH
M8005C208       | Mean of cell-wide RSSI on PUCCH
M8005C209       | Cell-wide RSSI on PUCCH level 1
M8005C210       | Cell-wide RSSI on PUCCH level 2
M8005C211       | Cell-wide RSSI on PUCCH level 3
M8005C212       | Cell-wide RSSI on PUCCH level 4
M8005C213       | Cell-wide RSSI on PUCCH level 5
M8005C214       | Cell-wide RSSI on PUCCH level 6
M8005C215       | Cell-wide RSSI on PUCCH level 7
M8005C216       | Cell-wide RSSI on PUCCH level 8
M8005C217       | Cell-wide RSSI on PUCCH level 9
M8005C218       | Cell-wide RSSI on PUCCH level 10
M8005C219       | Cell-wide RSSI on PUCCH level 11
M8005C220       | Cell-wide RSSI on PUCCH level 12
M8005C221       | Cell-wide RSSI on PUCCH level 13
M8005C222       | Cell-wide RSSI on PUCCH level 14
M8005C223       | Cell-wide RSSI on PUCCH level 15
M8005C224       | Cell-wide RSSI on PUCCH level 16
M8005C225       | Cell-wide RSSI on PUCCH level 17
M8005C226       | Cell-wide RSSI on PUCCH level 18
M8005C227       | Cell-wide RSSI on PUCCH level 19
M8005C228       | Cell-wide RSSI on PUCCH level 20
M8005C229       | Cell-wide RSSI on PUCCH level 21
M8005C230       | Cell-wide RSSI on PUCCH level 22
M8005C231       | Minimum of cell-wide RSSI on PUSCH
M8005C232       | Maximum of cell-wide RSSI on PUSCH
M8005C233       | Mean of cell-wide RSSI on PUSCH
M8005C234       | Cell-wide RSSI on PUSCH level 1
M8005C235       | Cell-wide RSSI on PUSCH level 2
M8005C236       | Cell-wide RSSI on PUSCH level 3
M8005C237       | Cell-wide RSSI on PUSCH level 4
M8005C238       | Cell-wide RSSI on PUSCH level 5
M8005C239       | Cell-wide RSSI on PUSCH level 6
M8005C240       | Cell-wide RSSI on PUSCH level 7
M8005C241       | Cell-wide RSSI on PUSCH level 8
M8005C242       | Cell-wide RSSI on PUSCH level 9
M8005C243       | Cell-wide RSSI on PUSCH level 10
M8005C244       | Cell-wide RSSI on PUSCH level 11
M8005C245       | Cell-wide RSSI on PUSCH level 12
M8005C246       | Cell-wide RSSI on PUSCH level 13
M8005C247       | Cell-wide RSSI on PUSCH level 14
```

M8005C248 | Cell-wide RSSI on PUSCH level 15
M8005C249 | Cell-wide RSSI on PUSCH level 16
M8005C250 | Cell-wide RSSI on PUSCH level 17
M8005C251 | Cell-wide RSSI on PUSCH level 18
M8005C252 | Cell-wide RSSI on PUSCH level 19
M8005C253 | Cell-wide RSSI on PUSCH level 20
M8005C254 | Cell-wide RSSI on PUSCH level 21
M8005C255 | Cell-wide RSSI on PUSCH level 22
M8005C256 | Minimum of cell-wide SINR on PUCCH
M8005C257 | Maximum of cell-wide SINR on PUCCH
M8005C258 | Mean of cell-wide SINR on PUCCH
M8005C259 | Cell-wide SINR on PUCCH level 1
M8005C260 | Cell-wide SINR on PUCCH level 2
M8005C261 | Cell-wide SINR on PUCCH level 3
M8005C262 | Cell-wide SINR on PUCCH level 4
M8005C263 | Cell-wide SINR on PUCCH level 5
M8005C264 | Cell-wide SINR on PUCCH level 6
M8005C265 | Cell-wide SINR on PUCCH level 7
M8005C266 | Cell-wide SINR on PUCCH level 8
M8005C267 | Cell-wide SINR on PUCCH level 9
M8005C268 | Cell-wide SINR on PUCCH level 10
M8005C269 | Cell-wide SINR on PUCCH level 11
M8005C270 | Cell-wide SINR on PUCCH level 12
M8005C271 | Cell-wide SINR on PUCCH level 13
M8005C272 | Cell-wide SINR on PUCCH level 14
M8005C273 | Cell-wide SINR on PUCCH level 15
M8005C274 | Cell-wide SINR on PUCCH level 16
M8005C275 | Cell-wide SINR on PUCCH level 17
M8005C276 | Cell-wide SINR on PUCCH level 18
M8005C277 | Cell-wide SINR on PUCCH level 19
M8005C278 | Cell-wide SINR on PUCCH level 20
M8005C279 | Cell-wide SINR on PUCCH level 21
M8005C280 | Cell-wide SINR on PUCCH level 22
M8005C281 | Minimum of cell-wide SINR on PUSCH
M8005C282 | Maximum of cell-wide SINR on PUSCH
M8005C283 | Mean of cell-wide SINR on PUSCH
M8005C284 | Cell-wide SINR on PUSCH level 1
M8005C285 | Cell-wide SINR on PUSCH level 2
M8005C286 | Cell-wide SINR on PUSCH level 3
M8005C287 | Cell-wide SINR on PUSCH level 4
M8005C288 | Cell-wide SINR on PUSCH level 5
M8005C289 | Cell-wide SINR on PUSCH level 6
M8005C290 | Cell-wide SINR on PUSCH level 7
M8005C291 | Cell-wide SINR on PUSCH level 8

```
M8005C292       | Cell-wide SINR on PUSCH level 9
M8005C293       | Cell-wide SINR on PUSCH level 10
M8005C294       | Cell-wide SINR on PUSCH level 11
M8005C295       | Cell-wide SINR on PUSCH level 12
M8005C296       | Cell-wide SINR on PUSCH level 13
M8005C297       | Cell-wide SINR on PUSCH level 14
M8005C298       | Cell-wide SINR on PUSCH level 15
M8005C299       | Cell-wide SINR on PUSCH level 16
M8005C300       | Cell-wide SINR on PUSCH level 17
M8005C301       | Cell-wide SINR on PUSCH level 18
M8005C302       | Cell-wide SINR on PUSCH level 19
M8005C303       | Cell-wide SINR on PUSCH level 20
M8005C304       | Cell-wide SINR on PUSCH level 21
M8005C305       | Cell-wide SINR on PUSCH level 22

-------------------------------------------------------------------------
Measurement Group: M8006
-------------------------------------------------------------------------
M8006C0         | EPS Bearer setup attempts
M8006C1         | EPS Bearer setup completions
M8006C2         | EPS Bearer setup failures due to Radio
M8006C3         | EPS Bearer setup failures due to
M8006C4         | EPS Bearer setup failures due to Radio
M8006C5         | EPS Bearer setup failures due to Other
M8006C6         | EPC initiated EPS Bearer Release
M8006C7         | EPC initiated EPS Bearer Release
M8006C8         | EPC initiated EPS Bearer Release
M8006C9         | EPC initiated EPS Bearer Release
M8006C10        | eNB initiated EPS Bearer Release
M8006C12        | Released E-RABs due to Radio Link
M8006C13        | eNB initiated EPS Bearer Release
M8006C14        | eNB initiated EPS Bearer Release
M8006C15        | eNB initiated EPS Bearer Release
M8006C17        | Initial EPS Bearer setup attempts for
M8006C18        | Initial EPS Bearer setup attempts for
M8006C26        | Additional EPS Bearer setup attempts
M8006C35        | Initial EPS Bearer setup completions
M8006C36        | Initial EPS Bearer setup completions
M8006C44        | Additional EPS Bearer setup
M8006C89        | EPC initiated EPS Bearer Release
M8006C98        | EPC initiated EPS Bearer Release
M8006C107       | EPC initiated EPS Bearer Release
M8006C116       | EPC initiated EPS Bearer Release
M8006C125       | eNB initiated EPS Bearer Release
```

```
M8006C134       | eNB initiated EPS Bearer Release
M8006C143       | eNB initiated EPS Bearer Release
M8006C152       | eNB initiated EPS Bearer Release
M8006C161       | eNB initiated EPS Bearer Release
M8006C162       | Initial EPS Bearer setup attempts for
M8006C163       | Initial EPS Bearer setup attempts for
M8006C164       | Initial EPS Bearer setup attempts for
M8006C165       | Additional EPS Bearer setup attempts
M8006C166       | Additional EPS Bearer setup attempts
M8006C167       | Additional EPS Bearer setup attempts
M8006C168       | Initial EPS Bearer setup completions
M8006C169       | Initial EPS Bearer setup completions
M8006C170       | Initial EPS Bearer setup completions
M8006C171       | Additional EPS Bearer setup
M8006C172       | Additional EPS Bearer setup
M8006C173       | Additional EPS Bearer setup
M8006C174       | Pre-empted GBR bearers
M8006C175       | Pre-empted non-GBR bearers
M8006C176       | Released active ERABs QCI1
M8006C177       | Released active ERABs QCI2
M8006C178       | Released active ERABs QCI3
M8006C179       | Released active ERABs QCI4
M8006C180       | Released active non GBR ERABs
M8006C181       | In-session activity time for QCI1
M8006C182       | In-session activity time for QCI2
M8006C183       | In-session activity time for QCI3
M8006C184       | In-session activity time for QCI4
M8006C185       | In-session activity time for non-GBR

-----------------------------------------------------------------------
Measurement Group: M8007
-----------------------------------------------------------------------
M8007C0         | Number of Data Radio Bearers
M8007C1         | Number of Data Radio Bearers
M8007C2         | Number of Data Radio Bearers failed to
M8007C3         | Number of released Data Radio
M8007C4         | Number of released Data Radio
M8007C5         | Number of released Data Radio
M8007C6         | Number of released Data Radio
M8007C7         | SRB1 setup attempts
M8007C8         | SRB1 setup completions
M8007C9         | SRB1 setup failures
M8007C10        | SRB2 setup attempts
M8007C11        | SRB2 setup completions
```

```
M8007C12        | SRB2 setup failures
M8007C13        | Radio Bearer Release requests

-------------------------------------------------------------------------
Measurement Group: M8008
-------------------------------------------------------------------------
M8008C0         | Rejected RRC Connection re-
M8008C1         | RRC Paging requests
M8008C2         | Discarded RRC Paging requests
M8008C3         | RRC Paging messages
M8008C4         | Number of RRC Connection Re-
M8008C5         | Number of successful RRC Connection
M8008C6         | RRC Connection Re-establishment
M8008C7         | RRC Connection Re-establishment
M8008C8         | RRC Connection Re-establishment
M8008C9         | RRC Connection Re-establishment
M8008C10        | Number of Report CGI Requests
M8008C11        | Number of successful CGI Reports
M8008C15        | Number of UEs attempted to redirect to
M8008C16        | Number of RRC Pagings for ETWS or

-------------------------------------------------------------------------
Measurement Group: M8009
-------------------------------------------------------------------------
M8009C0         | Total not started Handover preparations
M8009C1         | Total Handover decisions
M8009C2         | Intra eNB Handover preparations
M8009C3         | Failed Intra eNB Handover
M8009C5         | Failed Intra-eNB Handover
M8009C6         | Attempted intra eNB HO
M8009C7         | Successful intra eNB HO
M8009C8         | Total intra eNB HO failures due to timer

-------------------------------------------------------------------------
Measurement Group: M8010
-------------------------------------------------------------------------
M8010C36        | UE Reported CQI Level 00
M8010C37        | UE Reported CQI Level 01
M8010C38        | UE Reported CQI Level 02
M8010C39        | UE Reported CQI Level 03
M8010C40        | UE Reported CQI Level 04
M8010C41        | UE Reported CQI Level 05
M8010C42        | UE Reported CQI Level 06
M8010C43        | UE Reported CQI Level 07
```

```
M8010C44        | UE Reported CQI Level 08
M8010C45        | UE Reported CQI Level 09
M8010C46        | UE Reported CQI Level 10
M8010C47        | UE Reported CQI Level 11
M8010C48        | UE Reported CQI Level 12
M8010C49        | UE Reported CQI Level 13
M8010C50        | UE Reported CQI Level 14
M8010C51        | UE Reported CQI Level 15
M8010C52        | CQI offset Min
M8010C53        | CQI offset Max
M8010C54        | CQI offset Mean
M8010C55        | MIMO Open Loop Diversity Mode
M8010C56        | MIMO Open Loop Spatial Multiplexing
M8010C57        | MIMO Closed Loop Single Codeword
M8010C58        | MIMO Closed Loop Double Codeword
M8010C59        | Open Loop MIMO mode switches
M8010C60        | Closed Loop MIMO mode switches
M8010C61        | PDCCH allocations for PDSCH
M8010C62        | PDCCH allocations for PDSCH

-------------------------------------------------------------------------
Measurement Group: M8011
-------------------------------------------------------------------------
M8011C12        | UL PRB utilization per TTI Level 1
M8011C13        | UL PRB utilization per TTI Level 2
M8011C14        | UL PRB utilization per TTI Level 3
M8011C15        | UL PRB utilization per TTI Level 4
M8011C16        | UL PRB utilization per TTI Level 5
M8011C17        | UL PRB utilization per TTI Level 6
M8011C18        | UL PRB utilization per TTI Level 7
M8011C19        | UL PRB utilization per TTI Level 8
M8011C20        | UL PRB utilization per TTI Level 9
M8011C21        | UL PRB utilization per TTI Level 10
M8011C22        | UL PRB utilization per TTI Min
M8011C23        | UL PRB utilization per TTI Max
M8011C24        | UL PRB utilization per TTI Mean
M8011C25        | DL PRB utilization per TTI Level 1
M8011C26        | DL PRB utilization per TTI Level 2
M8011C27        | DL PRB utilization per TTI Level 3
M8011C28        | DL PRB utilization per TTI Level 4
M8011C29        | DL PRB utilization per TTI Level 5
M8011C30        | DL PRB utilization per TTI Level 6
M8011C31        | DL PRB utilization per TTI Level 7
M8011C32        | DL PRB utilization per TTI Level 8
```

```
M8011C33        | DL PRB utilization per TTI Level 9
M8011C34        | DL PRB utilization per TTI Level 10
M8011C35        | DL PRB utilisation per TTI Min
M8011C36        | DL PRB utilisation per TTI Max
M8011C37        | DL PRB utilisation per TTI Mean
M8011C38        | CCEs available PDCCH scheduling
M8011C39        | AGG1 used PDCCH
M8011C40        | AGG2 used PDCCH
M8011C41        | AGG4 used PDCCH
M8011C42        | AGG8 used PDCCH
M8011C43        | AGG1 blocked PDCCH
M8011C44        | AGG2 blocked PDCCH
M8011C45        | AGG4 blocked PDCCH
M8011C46        | AGG8 blocked PDCCH
M8011C47        | PRB used UL total
M8011C49        | PRB used PUCCH
M8011C50        | PRB used PUSCH
M8011C51        | PRB used DL total
M8011C54        | PRB used PDSCH
M8011C55        | Average UEs per UL TTI
M8011C56        | Max UEs per UL TTI
M8011C57        | Average UEs per DL TTI
M8011C58        | Max UEs per DL TTI
M8011C59        | Number of subframes with 1 OFDM
M8011C60        | Number of subframes with 2 OFDM
M8011C61        | Number of subframes with 3 OFDM
M8011C62        | Average number of UEs configured for
M8011C63        | Number of UL grants for TTI Bundling
M8011C64        | Short UE retention time in TTI Bundling
M8011C65        | Medium UE retention time in TTI
M8011C66        | Long UE retention time in TTI Bundling
M8011C67        | Number of SCell configuration attempts
M8011C68        | Number of successful SCell
M8011C69        | High cell load Indicator for Load

-------------------------------------------------------------------
Measurement Group: M8012
-------------------------------------------------------------------
M8012C0         | Total volume of original transmissions
M8012C1         | Total volume of correct received UL-
M8012C2         | Total volume of retransmissions of DL-
M8012C3         | Total volume of correct non-duplicate
M8012C12        | RLC data volume on DL DCCH
M8012C16        | RLC data volume on UL DCCH
```

```
M8012C17        | RLC PDU volume received
M8012C18        | RLC PDU volume transmitted
M8012C19        | PDCP SDU data volume on eUu
M8012C20        | PDCP SDU data volume on eUu
M8012C21        | PDCP Throughput UL Min
M8012C22        | PDCP Throughput UL Max
M8012C23        | PDCP Throughput UL Mean
M8012C24        | PDCP Throughput DL Min
M8012C25        | PDCP Throughput DL Max
M8012C26        | PDCP Throughput DL Mean
M8012C27        | MAC PDU volume PDSCH MCS0
M8012C28        | MAC PDU volume PDSCH MCS1
M8012C29        | MAC PDU volume PDSCH MCS2
M8012C30        | MAC PDU volume PDSCH MCS3
M8012C31        | MAC PDU volume PDSCH MCS4
M8012C32        | MAC PDU volume PDSCH MCS5
M8012C33        | MAC PDU volume PDSCH MCS6
M8012C34        | MAC PDU volume PDSCH MCS7
M8012C35        | MAC PDU volume PDSCH MCS8
M8012C36        | MAC PDU volume PDSCH MCS9
M8012C37        | MAC PDU volume PDSCH MCS10
M8012C38        | MAC PDU volume PDSCH MCS11
M8012C39        | MAC PDU volume PDSCH MCS12
M8012C40        | MAC PDU volume PDSCH MCS13
M8012C41        | MAC PDU volume PDSCH MCS14
M8012C42        | MAC PDU volume PDSCH MCS15
M8012C43        | MAC PDU volume PDSCH MCS16
M8012C44        | MAC PDU volume PDSCH MCS17
M8012C45        | MAC PDU volume PDSCH MCS18
M8012C46        | MAC PDU volume PDSCH MCS19
M8012C47        | MAC PDU volume PDSCH MCS20
M8012C48        | MAC PDU volume PUSCH MCS0
M8012C49        | MAC PDU volume PUSCH MCS1
M8012C50        | MAC PDU volume PUSCH MCS2
M8012C51        | MAC PDU volume PUSCH MCS3
M8012C52        | MAC PDU volume PUSCH MCS4
M8012C53        | MAC PDU volume PUSCH MCS5
M8012C54        | MAC PDU volume PUSCH MCS6
M8012C55        | MAC PDU volume PUSCH MCS7
M8012C56        | MAC PDU volume PUSCH MCS8
M8012C57        | MAC PDU volume PUSCH MCS9
M8012C58        | MAC PDU volume PUSCH MCS10
M8012C59        | MAC PDU volume PUSCH MCS11
M8012C60        | MAC PDU volume PUSCH MCS12
```

```
M8012C61        | MAC PDU volume PUSCH MCS13
M8012C62        | MAC PDU volume PUSCH MCS14
M8012C63        | MAC PDU volume PUSCH MCS15
M8012C64        | MAC PDU volume PUSCH MCS16
M8012C65        | MAC PDU volume PUSCH MCS17
M8012C66        | MAC PDU volume PUSCH MCS18
M8012C67        | MAC PDU volume PUSCH MCS19
M8012C68        | MAC PDU volume PUSCH MCS20
M8012C69        | MAC SDU volume on UL-CCCH
M8012C70        | MAC SDU volume on UL-DCCH
M8012C71        | MAC SDU volume on UL-DTCH
M8012C74        | MAC SDU volume on DL-CCCH
M8012C75        | MAC SDU volume on DL-DCCH
M8012C76        | MAC SDU volume on DL-DTCH
M8012C77        | RRC UL volume
M8012C78        | RRC DL volume
M8012C79        | RLC SDU UL-DTCH volume
M8012C80        | RLC SDU DL-DTCH volume
M8012C81        | MAC PDU volume PDSCH MCS21
M8012C82        | MAC PDU volume PDSCH MCS22
M8012C83        | MAC PDU volume PDSCH MCS23
M8012C84        | MAC PDU volume PDSCH MCS24
M8012C85        | MAC PDU volume PDSCH MCS25
M8012C86        | MAC PDU volume PDSCH MCS26
M8012C87        | MAC PDU volume PDSCH MCS27
M8012C88        | MAC PDU volume PDSCH MCS28
M8012C89        | Number of TTIs in UL with at least one
M8012C90        | Number of TTIs in DL with at least one
M8012C91        | IP Throughput volume in UL for QCI 1
M8012C92        | IP Throughput time in UL for QCI 1
M8012C93        | IP Throughput volume in UL for QCI 2
M8012C94        | IP Throughput time in UL for QCI 2
M8012C95        | IP Throughput volume in UL for QCI 3
M8012C96        | IP Throughput time in UL for QCI 3
M8012C97        | IP Throughput volume in UL for QCI 4
M8012C98        | IP Throughput time in UL for QCI 4
M8012C99        | IP Throughput volume in UL for QCI 5
M8012C100       | IP Throughput time in UL for QCI 5
M8012C101       | IP Throughput volume in UL for QCI 6
M8012C102       | IP Throughput time in UL for QCI 6
M8012C103       | IP Throughput volume in UL for QCI 7
M8012C104       | IP Throughput time in UL for QCI 7
M8012C105       | IP Throughput volume in UL for QCI 8
M8012C106       | IP Throughput time in UL for QCI 8
```

```
M8012C107       | IP Throughput volume in UL for QCI 9
M8012C108       | IP Throughput time in UL for QCI 9
M8012C116       | PDCP UL throughput Mean for QCI 1
M8012C117       | IP Throughput volume in DL for QCI 1
M8012C118       | IP Throughput time in DL for QCI 1
M8012C119       | IP Throughput volume in DL for QCI 2
M8012C120       | IP Throughput time in DL for QCI 2
M8012C121       | IP Throughput volume in DL for QCI 3
M8012C122       | IP Throughput time in DL for QCI 3
M8012C123       | IP Throughput volume in DL for QCI 4
M8012C124       | IP Throughput time in DL for QCI 4
M8012C125       | IP Throughput volume in DL for QCI 5
M8012C126       | IP Throughput time in DL for QCI 5
M8012C127       | IP Throughput volume in DL for QCI 6
M8012C128       | IP Throughput time in DL for QCI 6
M8012C129       | IP Throughput volume in DL for QCI 7
M8012C130       | IP Throughput time in DL for QCI 7
M8012C131       | IP Throughput volume in DL for QCI 8
M8012C132       | IP Throughput time in DL for QCI 8
M8012C133       | IP Throughput volume in DL for QCI 9
M8012C134       | IP Throughput time in DL for QCI 9
M8012C143       | PDCP DL throughput Mean for QCI 1
M8012C144       | MAC PDU volume PUSCH MCS21
M8012C145       | MAC PDU volume PUSCH MCS22
M8012C146       | MAC PDU volume PUSCH MCS23
M8012C147       | MAC PDU volume PUSCH MCS24
M8012C151       | PCell RLC data volume in DL via SCell

-----------------------------------------------------------------------
Measurement Group: M8013
-----------------------------------------------------------------------
M8013C5         | Signaling Connection Establishment
M8013C6         | Signaling Connection Establishment
M8013C7         | Signaling Connection Establishment
M8013C8         | Signaling Connection Establishment
M8013C9         | EPC initiated transitions to ECM-IDLE
M8013C10        | EPC initiated transitions to ECM-IDLE
M8013C11        | EPC initiated transitions to ECM-IDLE
M8013C12        | EPC initiated transitions to ECM-IDLE
M8013C13        | eNB initiated transitions to ECM-IDLE
M8013C15        | eNB initiated transitions to ECM-IDLE
M8013C16        | eNB initiated transitions to ECM-IDLE
M8013C17        | Signaling Connection Establishment
M8013C18        | Signaling Connection Establishment
```

```
M8013C19        | Signaling Connection Establishment
M8013C20        | Signaling Connection Establishment
M8013C21        | Number of Signaling Connection
M8013C24        | Subframes of DRX Active UE
M8013C25        | Subframes of DRX Sleep UE
M8013C26        | Signaling Connection Establishment
M8013C27        | Signaling Connection Establishment
M8013C28        | Pre-empted UE contexts due to pre-

-------------------------------------------------------------------------
Measurement Group: M8014
-------------------------------------------------------------------------
M8014C0         | Inter eNB Handover preparations
M8014C2         | Failed Inter eNB Handover
M8014C3         | Failed Inter eNB Handover
M8014C5         | Failed Inter-eNB Handover
M8014C6         | Attempted inter eNB HO
M8014C7         | Successful inter eNB HO
M8014C8         | Number of Inter eNB Handover failures
M8014C14        | Inter eNB S1-Handover preparations
M8014C15        | Failed Inter eNB S1-Handover
M8014C16        | Failed Inter eNB S1-Handover
M8014C17        | Failed Inter-eNB S1-Handover
M8014C18        | Attempted inter eNB S1-HO
M8014C19        | Successful inter eNB S1-HO
M8014C20        | Inter eNB S1-HO failures due to timer

-------------------------------------------------------------------------
Measurement Group: M8015
-------------------------------------------------------------------------
M8015C0         | Failed Intra eNB HO preparations per
M8015C1         | Intra eNB HO attempts per neighbor
M8015C2         | Intra eNB HO successes per neighbor
M8015C5         | Number of failed Inter eNB Handover
M8015C6         | Failed Inter eNB Handover
M8015C7         | Failed Inter eNB Handover
M8015C8         | Number of Inter eNB Handover
M8015C9         | Number of successful Inter eNB
M8015C10        | Number of Inter eNB Handover failures
M8015C15        | Intra eNB HO failures per neighbor cell
M8015C16        | Number of Late Handover per neighbor
M8015C17        | Number of Early Handover Type1 per
M8015C18        | Number of Early Handover Type2 per
```

```
-------------------------------------------------------------------------
Measurement Group: M8016
-------------------------------------------------------------------------
M8016C11        | CS Fallback attempts with redirection
M8016C12        | CS Fallback attempts (UE in
M8016C13        | CS Fallback attempts for emergency
M8016C14        | Inter System Handover preparations
M8016C15        | Failed Inter System Handover
M8016C16        | Failed Inter System Handover
M8016C17        | Failed Inter-System Handover
M8016C21        | Inter System Handover attempts
M8016C23        | Successful Inter System Handover
M8016C25        | Failed Inter System Handover attempts
M8016C26        | Number of NACC from LTE to GSM
M8016C29        | Inter System Handover attempts to
M8016C30        | Successful Inter System Handover
M8016C31        | Failed Inter System Handover attempts
M8016C32        | CS Fallback attempts to UTRAN with
M8016C33        | Inter System Handover attempts to
M8016C34        | Successful Inter System Handover
M8016C35        | Failed Inter System Handover attempts

-------------------------------------------------------------------------
Measurement Group: M8017
-------------------------------------------------------------------------
M8017C4         | Failed Inter System Handover
M8017C5         | Failed Inter System Handover
M8017C6         | Failed Inter System Handover
M8017C7         | Inter System Handover attempts to
M8017C8         | Successful Inter System Handover
M8017C9         | Failed Inter System Handover attempts
M8017C10        | Inter System Handover preparations
M8017C11        | Inter System Handover attempts to
M8017C12        | Successful Inter System Handover
M8017C13        | Failed Inter System Handover attempts

-------------------------------------------------------------------------
Measurement Group: M8018
-------------------------------------------------------------------------
M8018C0         | Active UE per eNB average
M8018C1         | Active UE per eNB max
M8018C4         | UE Contexts released due to S1 Reset
M8018C5         | Active UE Contexts released due to S1
M8018C6         | UE Contexts released due to S1 Reset
```

```
M8018C7         | Active UE Contexts released due to S1
M8018C8         | Sum of Active UEs per eNB
M8018C9         | Denominator for Active UEs per eNB

-----------------------------------------------------------------------
Measurement Group: M8019
-----------------------------------------------------------------------
M8019C0         | Number of NACC from LTE to GSM
M8019C3         | Inter System Handover attempts to
M8019C4         | Successful Inter System Handover
M8019C5         | Failed Inter System Handover attempts

-----------------------------------------------------------------------
Measurement Group: M8020
-----------------------------------------------------------------------
M8020C0         | Number of cell state changes to "cell is
M8020C1         | Number of cell state changes to "cell is
M8020C2         | Number of cell state changes to "cell is
M8020C3         | Samples when the cell is available
M8020C4         | Samples when the cell is planned
M8020C5         | Samples when the cell is unplanned
M8020C6         | Cell availability denominator

-----------------------------------------------------------------------
Measurement Group: M8021
-----------------------------------------------------------------------
M8021C0         | Number of inter frequency Handover
M8021C1         | Number of inter frequency Handover
M8021C2         | Number of successful inter-frequency
M8021C3         | Number of successful inter-frequency
M8021C4         | Number of Handover failures per cause
M8021C5         | Number of Handover failures per cause
M8021C6         | Handover preparations for IMS
M8021C9         | Failed Handover preparations for IMS
M8021C12        | Handover attempts for IMS emergency
M8021C15        | Successful Handover completions for
M8021C18        | Number of Handover attempts for UEs
M8021C19        | Number of Handover completions for
M8021C20        | Number of Late Handover
M8021C21        | Number of Early Handover Type1
M8021C22        | Number of Early Handover Type2
M8021C23        | Number of load balancing Handover
M8021C24        | Number of successful load balancing
```

```
-------------------------------------------------------------------------
Measurement Group: M8022
-------------------------------------------------------------------------
M8022C0         | Number of X2 Setup attempts
M8022C1         | Number of failed X2 Setup attempts

-------------------------------------------------------------------------
Measurement Group: M8023
-------------------------------------------------------------------------
M8023C0         | PDCP SDU data volume on eUu
M8023C1         | PDCP SDU data volume on eUu
M8023C2         | PDCP SDU data volume on eUu
M8023C3         | PDCP SDU data volume on eUu
M8023C4         | PDCP SDU data volume on eUu
M8023C5         | PDCP SDU data volume on eUu
M8023C6         | PDCP SDU data volume on eUu
M8023C7         | PDCP SDU data volume on eUu
M8023C8         | PDCP SDU data volume on eUu
M8023C9         | PDCP SDU data volume on eUu
M8023C10        | PDCP SDU data volume on eUu
M8023C11        | PDCP SDU data volume on eUu
M8023C12        | Average Number of active UE of
M8023C13        | Average Number of active UE of
M8023C14        | Average Number of active UE of
M8023C15        | Average Number of active UE of
M8023C16        | Average Number of active UE of
M8023C17        | Average Number of active UE of
M8023C18        | Average Number of active UE of
M8023C19        | Average Number of active UE of
M8023C20        | Average Number of active UE of
M8023C21        | Average number of active UEs of QCI
M8023C22        | Average number of active UEs of QCI
M8023C23        | Average number of active UEs of QCI
M8023C24        | Average number of active UEs of QCI
M8023C25        | Average number of active UEs of QCI
M8023C26        | Average number of active UEs of QCI
M8023C27        | Number of QCI Counter Group
M8023C28        | Average number of active UEs of
M8023C29        | Average number of active UEs of
M8023C30        | Average number of active UEs of
M8023C31        | Average number of active UEs of
M8023C32        | Average number of active UEs of
M8023C33        | Average number of active UEs of
M8023C34        | Average number of active UEs of
```

```
M8023C35        | Average number of active UEs of

-------------------------------------------------------------------------
Measurement Group: M5112
-------------------------------------------------------------------------
M51120C0        | ifInPackets15
M51120C1        | ifInOctets15
M51120C2        | ifOutPackets15
M51120C3        | ifOutOctets15
M51120C4        | ifInErrors15
M51120C5        | ifOutDroppedPackets
M51120C6        | ifOutDroppedOctets
M51121C0        | ifOctets_EF
M51121C1        | ifPackets_EF
M51121C2        | ifDroppedOctets_EF
M51121C3        | ifDroppedPackets_EF
M51121C4        | ifOctets_AF4
M51121C5        | ifPackets_AF4
M51121C6        | ifDroppedOctets_AF4
M51121C7        | ifDroppedPackets_AF4
M51121C8        | ifOctets_AF3
M51121C9        | ifPackets_AF3
M51121C10       | ifDroppedOctets_AF3
M51121C11       | ifDroppedPackets_AF3
M51121C12       | ifOctets_AF2
M51121C13       | ifPackets_AF2
M51121C14       | ifDroppedOctets_AF2
M51121C15       | ifDroppedPackets_AF2
M51121C16       | ifOctets_AF1
M51121C17       | ifPackets_AF1
M51121C18       | ifDroppedOctets_AF1
M51121C19       | ifDroppedPackets_AF1
M51121C20       | ifOctets_BE
M51121C21       | ifPackets_BE
M51121C22       | ifDroppedOctets_BE
M51121C23       | ifDroppedPackets_BE
M51121C24       | ifRxOctets_EF
M51121C25       | ifRxPackets_EF
M51121C26       | ifRxOctets_AF4
M51121C27       | ifRxPackets_AF4
M51121C28       | ifRxOctets_AF3
M51121C29       | ifRxPackets_AF3
M51121C30       | ifRxOctets_AF2
M51121C31       | ifRxPackets_AF2
```

M51121C32	ifRxOctets_AF1
M51121C33	ifRxPackets_AF1
M51121C34	ifRxOctets_BE
M51121C35	ifRxPackets_BE
M51123C0	EthIfInOcts_15
M51123C1	EthIfOutOcts_15
M51123C2	EthIfInPkt_15
M51123C3	EthIfOutPkt_15
M51123C4	EthIfInPktErr_15
M51123C8	EthIfInDiscRateLimiting
M51123C16	ethIfInVlanMismatch
M51123C17	ethIfOutDiscShaping_Q5
M51123C18	ethIfOutDiscShaping_Q4
M51123C19	ethIfOutDiscShaping_Q3
M51123C20	ethIfOutDiscShaping_Q2
M51123C21	ethIfOutDiscShaping_Q1
M51123C22	ethIfOutDiscShaping_Q6
M51123C23	EthIfInBlocksDiscRateLimiting
M51125C0	Protected_ESPFramesTx
M51125C1	Protected_ESPFramesRx
M51125C2	Discarded_ESPFramesTx
M51125C3	Discarded_ESPFramesRx
M51125C4	Bypassed_FramesTx
M51125C5	Bypassed_FramesRx
M51126C0	ipRmDroppedPacketsRateLimiting
M51126C1	ipRmDroppedPacketsFiltering
M51127C0	ifInPackets15
M51127C1	ifInOctets15
M51127C2	ifOutPackets15
M51127C3	ifOutOctets15
M51127C4	ifInErrors15
M51127C5	ifOutDroppedPackets
M51127C6	ifOutDroppedOctets
M51128C0	ifOctets_EF
M51128C1	ifPackets_EF
M51128C2	ifDroppedOctets_EF
M51128C3	ifDroppedPackets_EF
M51128C4	ifOctets_AF4
M51128C5	ifPackets_AF4
M51128C6	ifDroppedOctets_AF4
M51128C7	ifDroppedPackets_AF4
M51128C8	ifOctets_AF3
M51128C9	ifPackets_AF3
M51128C10	ifDroppedOctets_AF3

```
M51128C11       | ifDroppedPackets_AF3
M51128C12       | ifOctets_AF2
M51128C13       | ifPackets_AF2
M51128C14       | ifDroppedOctets_AF2
M51128C15       | ifDroppedPackets_AF2
M51128C16       | ifOctets_AF1
M51128C17       | ifPackets_AF1
M51128C18       | ifDroppedOctets_AF1
M51128C19       | ifDroppedPackets_AF1
M51128C20       | ifOctets_BE
M51128C21       | ifPackets_BE
M51128C22       | ifDroppedOctets_BE
M51128C23       | ifDroppedPackets_BE
M51128C24       | ifRxOctets_EF
M51128C25       | ifRxPackets_EF
M51128C26       | ifRxOctets_AF4
M51128C27       | ifRxPackets_AF4
M51128C28       | ifRxOctets_AF3
M51128C29       | ifRxPackets_AF3
M51128C30       | ifRxOctets_AF2
M51128C31       | ifRxPackets_AF2
M51128C32       | ifRxOctets_AF1
M51128C33       | ifRxPackets_AF1
M51128C34       | ifRxOctets_BE
M51128C35       | ifRxPackets_BE
M51129C0        | rxEthernetPackets
M51129C1        | rxEhernetOctets
M51129C2        | txEthernetPackets
M51129C3        | txEthernetOctets

-------------------------------------------------------------------------
Measurement Group: M5113
-------------------------------------------------------------------------
M51130C0        | rxEthernetPackets
M51130C1        | rxEhernetOctets
M51130C2        | txEthernetPackets
M51130C3        | txEthernetOctets
M51130C4        | EthIfInUnknownVLAN_15
M51131C0        | ethIfOutDiscShaping
M51132C0        | avgRTT_15Min
M51132C1        | maxRTT_15Min
M51132C2        | minRTT_15Min
M51132C3        | lostTwampMessages
M51132C4        | txTwampMessages
```

```
M51136C0        | tacSuccessfulGbrNormal
M51136C1        | tacSuccessfulGbrHandover
M51136C2        | tacSuccessfulGbrEmergency
M51136C3        | tacRejectedGbrNormal
M51136C4        | tacRejectedGbrHandover
M51136C5        | tacRejectedGbrEmergency
M51137C0        | topRxFreqSyncMsg
M51137C1        | topMinFreqSyncError
M51137C2        | topAvgFreqSyncError
M51137C3        | topMaxFreqSyncError
```

Related Documentation

- **Grafana Integration** - Building dashboards using these counters
- **Data Retention Policy** - Managing counter data lifecycle
- **Runtime Configuration Guide** - Configuring data collection intervals
- **AirScale Configuration** - Base station setup and registration

PM Data Collection Guide

Overview

The PM Data Collection page allows you to manage which Performance Metrics (PM) counters are stored in InfluxDB. Nokia AirScale base stations report over **22,000 unique PM counters**, but storing all of them is neither practical nor necessary for most use cases.

This guide explains how to select which counters to collect based on your monitoring requirements.

Quick Start

Accessing the PM Data Collection Page

1. Navigate to the Control Panel: `https://localhost:9443`
2. Click on **Data Filters** in the navigation menu
3. View and manage PM counter collection settings

Understanding the Interface

The page is divided into two main sections:

Section	Description
Stored PM Data (Left)	Counters currently being collected and stored in InfluxDB
Available Counters (Right)	All 22,000+ counters available to add to your collection

---

Counter Categories

PM counters are categorized by their code prefix, which indicates the technology and function:

Category	Code Prefix	Count	Description
LTE	M8xxx	~5,900	LTE L1/L2/L3 counters (ERAB, RRC, handover, etc.)
WCDMA	M5xxx	~885	3G WCDMA counters (MAC layer, CQI, HSDPA)
5G-NR	M55xxx	~14,500	5G NR counters (massive MIMO, beamforming, etc.)
5G-Mobility	M51xxx	~500	5G mobility and measurements
5G-Common	M40xxx	~250	5G common/shared counters

Default Counters

On first startup, sensible defaults are loaded from `priv/pm_counters.csv`. These defaults include essential counters for:

- **Energy**: Power consumption monitoring
- **Data Volume**: Traffic volume metrics
- **Availability**: Cell availability statistics
- **Accessibility**: RRC connection success/failure
- **PRB**: Physical Resource Block utilization
- **Throughput**: UL/DL throughput metrics
- **RRC**: RRC connection statistics
- **ERAB**: E-RAB setup and release counters
- **PDCP**: PDCP layer metrics
- **Handover**: Inter-cell handover statistics
- **Interference**: UL interference measurements

Managing Counters

Adding Counters

1. Use the **search box** or **category filter** in the "Available Counters" section
2. Click on rows to select counters (checkbox will appear checked)
3. Use **Select All** to select all visible counters
4. Click **Add Selected** to move them to the stored collection

Removing Counters

1. In the "Stored PM Data" section, select counters to remove
2. Click **Remove Selected** to stop collecting those counters

Filtering and Searching

Both sections support:

- **Text search**: Filter by counter ID or description
- **Category filter**: Show only counters from a specific category (LTE, 5G-NR, etc.)

Resetting to Defaults

Click **Reset to Defaults** to restore the original counter list from `priv/pm_counters.csv`. This will remove any custom additions.

Persistence

Changes to your PM counter selection are:

1. **Persisted to disk** in `priv/pm_filters.etf`
2. **Survive application restarts**
3. **Take effect immediately** (no restart required)

The InfluxDB batch writer is notified of changes and immediately starts/stops collecting the affected counters.

Storage Considerations

Why Not Collect Everything?

Collecting all 22,000+ counters would result in:

Scenario	Impact
Storage	~100-500 GB/month per site (depending on collection interval)
Write Load	Significant InfluxDB write pressure
Query Performance	Slower dashboard queries due to data volume
Cost	Higher storage and compute costs

Recommended Approach

1. **Start with defaults**: The pre-configured counters cover most common monitoring needs
2. **Add as needed**: When building new dashboards, add specific counters you need
3. **Review periodically**: Remove counters no longer being used

---

Counter Reference

Finding Counter Descriptions

The "Available Counters" section shows Nokia's official description for each counter. Use the search function to find counters by:

- **Counter ID** (e.g., `M8012C23`)
- **Description keywords** (e.g., `throughput`, `handover`, `RSRP`)

Common Counter Examples

Counter	Category	Description
M8012C23	LTE	Average UL Throughput per cell
M8012C26	LTE	Average DL Throughput per cell
M8001C2	LTE	PDCP SDU Delay DL Mean
M8011C24	LTE	UL PRB Utilization
M8011C37	LTE	DL PRB Utilization
M8013C17	LTE	RRC Connected Users
M8020C3	LTE	Handover Success
M40001C0	5G	Energy Consumption

Configuration Files

pm_counters.csv

Default counters loaded on first startup:

```
# Format: counter,category,description
M8012C23,Throughput,Average Uplink Throughput
M8012C26,Throughput,Average Downlink Throughput
M8001C2,Availability,Cell Availability
...
```

Location: `priv/pm_counters.csv`

pm_metrics.csv

Complete reference of all available counters:

```
# Format: PM_Code,Category,Description
M8000C6,LTE,S1_SETUP_ATT
M8000C7,LTE,S1_SETUP_SUCC
...
```

Location: `priv/pm_metrics.csv`

Troubleshooting

Counters Not Being Collected

1. Verify the counter is in "Stored PM Data" (left side)
2. Check that the eNodeB is pushing PM data (see InfluxDB Status page)
3. Verify the counter ID matches exactly (case-sensitive)

Changes Not Taking Effect

1. Filter changes are applied immediately to the batch writer
2. **New data only appears after the next PM push from the eNodeB** (typically every 15 minutes)
3. Check application logs for `[PmFilterStore]` errors
4. Verify disk is writable for persistence file

Missing Counter Descriptions

1. Counter descriptions come from `priv/pm_metrics.csv`
2. Ensure this file is present and properly formatted
3. Check for UTF-8 encoding issues

Related Documentation

- Data Retention Policy - How long PM data is kept
- Grafana Integration - Building dashboards with PM data
- InfluxDB Queries - Querying PM data

Access Points

- **PM Data Collection**: `https://localhost:9443/nokia/pm-filters`
- **Data Retention**: `https://localhost:9443/nokia/retention`
- **InfluxDB Status**: `https://localhost:9443/nokia/influx`

RAN Monitor Runtime Configuration Guide

Understanding config/runtime.exs

Table of Contents

Overview

The `config/runtime.exs` file is the primary configuration file for RAN Monitor. It's evaluated at runtime (when the application starts), allowing you to configure all aspects of the system's behavior.

What Gets Configured:

- Database connections (MySQL)
- Web server endpoints and ports
- Nokia base station details
- InfluxDB time-series database
- Logging behavior
- Security credentials

File Location:

```
config/runtime.exs
```

Who Should Use This Guide

Important: All RAN Monitor configuration is **performed by Omnitouch** as part of the initial deployment and ongoing support. This guide is provided for:

- **Advanced users** who want to understand the system configuration
- **Self-managed deployments** where customers maintain their own configuration
- **Troubleshooting** and understanding how the system is configured
- **Custom deployments** with specific requirements

If you are an Omnitouch-managed customer, contact Omnitouch support for any configuration changes.

For understanding what data is being collected, see Nokia Counter Reference. For dashboard creation, see Grafana Integration.

Database Configuration

MySQL/MariaDB Connection

```
config :ran_monitor, RanMonitor.Repo,
  username: "omnitouch",
  password: "omnitouch2024",
  hostname: "localhost",
  database: "ran_monitor",
  stacktrace: true,
  show_sensitive_data_on_connection_error: true,
  pool_size: 10
```

Purpose: Configures the connection to the MySQL database used for session state management and operational data.

Parameters Explained

username (String)

- Database user account
- Current value: `"omnitouch"`
- **Usage:** Must have CREATE, SELECT, INSERT, UPDATE, DELETE privileges
- **Security:** Consider using a dedicated user with minimum required privileges

password (String)

- Database password for authentication
- Current value: `"omnitouch2024"`
- **Security:** Should be stored in environment variables in production
- **Recommendation:** Use strong, unique passwords

hostname (String)

- Database server address
- Current value: `"localhost"`
- **Options:**
  - `"localhost"` - Database on same machine
  - `"127.0.0.1"` - TCP connection to local machine
  - `"10.179.2.135"` - Remote database server IP
  - `"db.example.com"` - Remote database hostname

database (String)

- Database name to use
- Current value: `"ran_monitor"`
- **Note:** Database must exist before starting RAN Monitor
- **Creation:** `CREATE DATABASE ran_monitor CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;`

stacktrace (Boolean)

- Include stacktraces in error messages
- Current value: `true`
- **Development:** `true` - Helps with debugging
- **Production:** `false` - Reduces log noise

show_sensitive_data_on_connection_error (Boolean)

- Show credentials in connection error messages
- Current value: `true`
- **Development:** `true` - Easier troubleshooting
- **Production:** `false` - Prevents credential exposure in logs

pool_size (Integer)

- Number of database connections to maintain
- Current value: `10`
- **Sizing Guide:**
  - 1-5 devices: `pool_size: 5`
  - 6-20 devices: `pool_size: 10`
  - 21-50 devices: `pool_size: 15`
  - 50+ devices: `pool_size: 20`
- **Formula:** Roughly 2 connections per base station + 5 for web UI

Web Endpoints

RAN Monitor runs multiple web servers for different purposes.

Main SOAP/API Endpoint

```
config :ran_monitor, RanMonitor.Web.Endpoint,
  http: [ip: {0, 0, 0, 0}, port: 8080],
  check_origin: false,
  secret_key_base:
"v5tOS1/QRonjwOky7adGGfkBbrJmiJyXhpesJy/jvSZhqLZkREV+rlo1/pR8lkbu",
  server: true
```

Purpose: Main endpoint for base station communication (SOAP interface for Nokia NE3S protocol).

ip (Tuple)

- Interface to bind to
- Current value: `{0, 0, 0, 0}` (all interfaces)
- **Options:**
  - `{0, 0, 0, 0}` - Listen on all network interfaces
  - `{127, 0, 0, 1}` - Listen only on localhost
  - `{10, 179, 2, 135}` - Listen on specific IP address

port (Integer)

- TCP port number
- Current value: `8080`
- **Note:** Base stations must be configured to send data to this port
- **Firewall:** Ensure port is open for base station IPs

check_origin (Boolean)

- Validate WebSocket/HTTP origin headers
- Current value: `false`
- **Explanation:** Set to `false` for SOAP API (not user-facing web UI)

secret_key_base (String)

- Cryptographic signing key for sessions
- Current value: 64-character random string

- **Generation:** `mix phx.gen.secret`
- **Security:** Keep this secret, never commit to public repositories
- **Impact:** Changing this invalidates all existing sessions

server (Boolean)

- Start the endpoint when application starts
- Current value: `true`
- **Always:** Should be `true` in runtime.exs

Control Panel Web UI

```
# Get HTTPS port from environment variable, default to 9443
https_port =
String.to_integer(System.get_env("CONTROL_PANEL_HTTPS_PORT") ||
"9443")

config :control_panel, ControlPanelWeb.Endpoint,
  url: [host: "0.0.0.0", port: https_port, scheme: "https"],
  https: [
    ip: {0, 0, 0, 0},
    port: https_port,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ]
```

Purpose: HTTPS endpoint for the web-based control panel UI.

Environment Variables:

- **CONTROL_PANEL_HTTPS_PORT** - HTTPS port number (default: 9443)
  - Set this environment variable to change the HTTPS port at runtime
  - Example: `export CONTROL_PANEL_HTTPS_PORT=8443`

url (Keyword list)

- External URL configuration
- **host:** `"0.0.0.0"` - Accept connections from any host

- **port:** Uses `https_port` variable (configurable via CONTROL_PANEL_HTTPS_PORT)
- **scheme:** `"https"` - Use HTTPS protocol

https (Keyword list)

- HTTPS server configuration
- **ip:** `{0, 0, 0, 0}` - Bind to all interfaces
- **port:** Uses `https_port` variable (must match url port)
- **keyfile:** Path to SSL private key
- **certfile:** Path to SSL certificate

SSL Certificate Files:

- Must be valid SSL/TLS certificates
- Self-signed certificates work for lab environments
- Production should use CA-signed certificates
- Generate self-signed:

```
openssl req -newkey rsa:2048 -nodes -keyout omnitouch.pem -x509
-days 365 -out omnitouch.crt
```

Nokia AirScale Webhook Endpoint

```
config :ran_monitor, RanMonitor.Web.Nokia.Airscale.Endpoint,
  url: [host: "0.0.0.0"],
  http: [ip: {0, 0, 0, 0}, port: 9076],
  server: true
```

Purpose: Receives real-time performance data from Nokia AirScale base stations.

port (Integer)

- Current value: `9076`
- **Note:** Must match the port configured in base station PMCADM (rTpmCollEntityPortNum)

- **Coordination:** This port must match what you configured in the Nokia WebLM Parameter Editor

---

Logger Configuration

```
config :logger,
  level: :info

config :logger, :console,
  format: "$time $metadata[$level] $message\n",
  metadata: [:request_id]
```

Log Level

level (Atom)

- Controls verbosity of logging
- Current value: `:info`
- **Options:**
  - `:debug` - Extremely verbose, all details
  - `:info` - Normal operations, recommended for production
  - `:warning` - Only warnings and errors
  - `:error` - Only errors

When to Use Each Level:

- **Development:** `:debug` - See all internal operations
- **Production:** `:info` - Balance between visibility and noise
- **Troubleshooting:** Temporarily set to `:debug`, then revert
- **Quiet Production:** `:warning` - Only alert on issues

Console Format

format (String)

- How log messages appear
- Current value: `"$time $metadata[$level] $message\n"`
- **Variables:**
  - `$time` - Timestamp
  - `$metadata` - Contextual information
  - `$level` - Log level (info, error, etc.)
  - `$message` - Actual log message

metadata (List of atoms)

- Additional context to include
- Current value: `[:request_id]`
- **request_id:** Tracks individual HTTP requests through the system

---

Nokia Integration

This section configures how RAN Monitor communicates with Nokia base stations.

```
config :ran_monitor,
  general: %{
    mcc: "505",
    mnc: "57"
  },
  nokia: %{
    ne3s: %{
      webhook_url: "http://10.5.198.200:9076/webhook",
      private_key: Path.join(Application.app_dir(:ran_monitor,
"priv"), "external/nokia/ne.key.pem"),
      public_key: Path.join(Application.app_dir(:ran_monitor,
"priv"), "external/nokia/ne.cert.der"),
      reregister_interval: 30
    },
    airscales: [
      %{
        address: "10.7.15.67",
        name: "ONS-Lab-Airscale",
        port: "8080",
        web_username: "Nemuadmin",
        web_password: "nemuuser"
      }
    ]
  }
```

General Settings

mcc (String)

- Mobile Country Code
- Current value: `"505"`
- **Usage:** Identifies the country for 3GPP networks
- **Format:** 3 digits
- **Reference:** ITU-T E.212

mnc (String)

- Mobile Network Code
- Current value: `"57"`

- **Usage:** Identifies the specific network operator
- **Format:** 2 or 3 digits

NE3S Configuration (Nokia NE3S Protocol)

webhook_url (String)

- URL where base stations send notifications
- Current value: `"http://10.5.198.200:9076/webhook"`
- **Format:** `http://<ran-monitor-ip>:<port>/webhook`
- **IP Address:** Must be IP address where RAN Monitor is running
- **Port:** Must match `RanMonitor.Web.Nokia.Airscale.Endpoint` port (9076)
- **Path:** Always `/webhook`

private_key (String - File path)

- Private key for manager authentication
- Current value: `priv/external/nokia/ne.key.pem`
- **Format:** PEM encoded private key
- **Security:** Keep this file secure, never share
- **Generation:** Provided by Nokia or generated with OpenSSL

public_key (String - File path)

- Public certificate for manager identity
- Current value: `priv/external/nokia/ne.cert.der`
- **Format:** DER encoded certificate
- **Usage:** Sent to base station during registration
- **Pair:** Must correspond to private_key

reregister_interval (Integer)

- How often to re-register with base stations (seconds)
- Current value: `30`
- **Explanation:** Sessions expire, periodic re-registration maintains connection

- **Range:** 30-300 seconds
- **Recommendation:** 30 seconds for reliable monitoring

AirScale Base Stations

airscales (List of maps)

- List of Nokia AirScale base stations to monitor
- Current value: One base station configured

Each base station entry requires:

address (String)

- IP address of the base station
- Current value: `"10.7.15.66"`
- **Format:** IPv4 address as string
- **Network:** Must be reachable from RAN Monitor server
- **Verification:** `ping 10.7.15.66` should succeed

name (String)

- Friendly name for identification
- Current value: `"ONS-Lab-Airscale"`
- **Usage:** Appears in Web UI, logs, and InfluxDB tags
- **Recommendation:** Use descriptive names (site codes, locations, etc.)
- **Examples:**
  - `"NYC-Site-A-BS1"`
  - `"LAX-Tower-Main"`
  - `"TestLab-Airscale-01"`

port (String)

- Management interface port on base station
- Current value: `"8080"`
- **Standard:** Nokia AirScale typically uses 8080
- **Verification:** Check base station documentation

- **Note:** Value is a string, not integer

web_username (String)

- Username for WebLM authentication
- Current value: `"Nemuadmin"`
- **Usage:** Used for API calls to manage base station
- **Privileges:** Should have configuration read/write access
- **Note:** Case-sensitive

web_password (String)

- Password for WebLM authentication
- Current value: `"nemuuser"`
- **Security:** Should be stored in environment variables in production
- **Rotation:** Change regularly according to security policy

Adding Multiple Base Stations

To monitor multiple base stations, add additional entries to the `airscales` list:

```
airscales: [
  %{
    address: "10.7.15.66",
    name: "ONS-Lab-Airscale",
    port: "8080",
    web_username: "Nemuadmin",
    web_password: "nemuuser"
  },
  %{
    address: "10.7.15.67",
    name: "Site-A-Tower-1",
    port: "8080",
    web_username: "admin",
    web_password: "password123"
  },
  %{
    address: "192.168.100.50",
    name: "Site-B-Indoor",
    port: "8080",
    web_username: "admin",
    web_password: "different_password"
  }
]
```

InfluxDB Configuration

```
config :ran_monitor, RanMonitor.InfluxDbConnection,
  auth: [
    username: "monitor",
    password: "sideunderTexasgalaxyview_61"
  ],
  database: "nokia-monitor",
  host: "10.179.2.135"
```

Purpose: Configures connection to InfluxDB time-series database for storing metrics, alarms, and configuration data.

Parameters Explained

auth (Keyword list)

- Authentication credentials for InfluxDB
- **username:** InfluxDB user account (`"monitor"`)
- **password:** InfluxDB password (`"sideunderTexasgalaxyview_61"`)
- **Note:** For InfluxDB 2.x, this might be an API token instead

database (String)

- Bucket/database name in InfluxDB
- Current value: `"nokia-monitor"`
- **InfluxDB 1.x:** Database name
- **InfluxDB 2.x:** Bucket name
- **Creation:** Must be created before starting RAN Monitor

```
# InfluxDB 1.x
influx -execute 'CREATE DATABASE "nokia-monitor"'

# InfluxDB 2.x
influx bucket create -n nokia-monitor -o your-org
```

host (String)

- InfluxDB server address
- Current value: `"10.179.2.135"`
- **Format:** IP address or hostname
- **Port:** Default InfluxDB port (8086) is assumed
- **Examples:**
  - `"localhost"` - Same server as RAN Monitor
  - `"10.179.2.135"` - Remote InfluxDB server
  - `"influxdb.example.com"` - Hostname

InfluxDB Connection Notes

Network Access:

- RAN Monitor must be able to reach InfluxDB server on port 8086
- Verify: `curl http://10.179.2.135:8086/ping`

Retention Policies:

- Set via Web UI Data Retention page
- Default: 30 days (720 hours)
- Can be customized per base station

Write Performance:

- InfluxDB receives writes every collection interval (60s default)
- Each base station generates hundreds of data points per interval
- Monitor InfluxDB disk space regularly

Configuration Best Practices

Security

1. Protect Sensitive Data

```
# Instead of hardcoded passwords:
password: "omnitouch2024"

# Use environment variables:
password: System.get_env("DB_PASSWORD") || "default_password"
```

2. Restrict File Permissions

```
chmod 600 config/runtime.exs
chown ran_monitor:ran_monitor config/runtime.exs
```

3. Never Commit Secrets

- Use `.gitignore` for runtime.exs if it contains secrets
- Use environment variables or secret management systems
- Rotate passwords regularly

Performance

1. Database Pool Sizing

- Monitor connection usage
- Increase pool_size if seeing connection timeout errors
- Each device needs ~2 connections during active polling

2. Collection Intervals

- Balance between data granularity and system load
- 60 second intervals work well for most deployments
- Shorter intervals (15s) for troubleshooting

3. InfluxDB Optimization

- Use retention policies to manage disk usage
- Monitor InfluxDB write performance
- Consider separate InfluxDB server for large deployments

Reliability

1. Network Configuration

- Use static IP addresses for all components
- Verify network routes between RAN Monitor and base stations
- Test connectivity before adding devices
- Configure firewall rules appropriately

2. Logging Strategy

- Development: `:debug` for detailed troubleshooting
- Production: `:info` for operational visibility
- Critical systems: Consider external log aggregation

3. Monitoring RAN Monitor

- Monitor the monitor (meta-monitoring)
- Watch for database connection errors
- Track InfluxDB write success rates
- Alert on base station disconnections

Maintenance

1. Configuration Changes

- Always backup runtime.exs before changes
- Test configuration in development first
- Document changes with comments
- Restart RAN Monitor after configuration changes

2. Adding Base Stations

```
# 1. Edit runtime.exs
vim config/runtime.exs

# 2. Validate Elixir syntax
elixir -c config/runtime.exs

# 3. Restart application
systemctl restart ran_monitor
```

3. Scaling Considerations

- Monitor resource usage (CPU, memory, network)
- Increase database pool size as device count grows
- Consider separate InfluxDB instance at 50+ devices
- Monitor disk space for both MySQL and InfluxDB

Example: Complete Configuration

Here's a complete example with multiple base stations and best practices applied:

```
import Config

#
============================================================================
# Database Configuration
#
============================================================================

config :ran_monitor, RanMonitor.Repo,
  username: System.get_env("DB_USERNAME") || "ran_monitor_user",
  password: System.get_env("DB_PASSWORD") || "change_this_password",
  hostname: System.get_env("DB_HOST") || "localhost",
  database: "ran_monitor",
  stacktrace: false,  # Production: hide stacktraces
  show_sensitive_data_on_connection_error: false,  # Production: hide
credentials
  pool_size: 15  # 6 base stations * 2 + 3 overhead

#
============================================================================
# Web Endpoints
#
============================================================================

config :ran_monitor, RanMonitor.Web.Endpoint,
  http: [ip: {0, 0, 0, 0}, port: 8080],
  check_origin: false,
  secret_key_base: System.get_env("SECRET_KEY_BASE") ||
"generate_with_mix_phx_gen_secret",
  server: true

config :control_panel, ControlPanelWeb.Endpoint,
  url: [host: "0.0.0.0", port: 9443, scheme: "https"],
  https: [
    ip: {0, 0, 0, 0},
    port: 9443,
    keyfile: "priv/cert/server.key",
    certfile: "priv/cert/server.crt"
  ]

config :ran_monitor, RanMonitor.Web.Nokia.Airscale.Endpoint,
  url: [host: "0.0.0.0"],
  http: [ip: {0, 0, 0, 0}, port: 9076],
```

```
  server: true

#
# Logger Configuration
#

config :logger,
  level: :info  # Production setting

config :logger, :console,
  format: "$time $metadata[$level] $message\n",
  metadata: [:request_id]

#
# Nokia Configuration
#

config :ran_monitor,
  general: %{
    mcc: "001",
    mnc: "001"
  },
  nokia: %{
    ne3s: %{
      webhook_url: "http://10.179.2.135:9076/webhook",
      private_key: Path.join(Application.app_dir(:ran_monitor, "priv"
"external/nokia/ne.key.pem"),
      public_key: Path.join(Application.app_dir(:ran_monitor, "priv")
"external/nokia/ne.cert.der"),
      reregister_interval: 30
    },
    airscales: [
      # Site A - Main Tower
      %{
        address: "10.7.15.66",
        name: "Site-A-Main-Tower",
        port: "8080",
        web_username: "admin",
        web_password: System.get_env("BS_SITE_A_PASSWORD") || "passwo
      },
```

```
    # Site A - Backup Tower
    %{
      address: "10.7.15.67",
      name: "Site-A-Backup-Tower",
      port: "8080",
      web_username: "admin",
      web_password: System.get_env("BS_SITE_A_PASSWORD") || "passwo
    },

    # Site B - Indoor
    %{
      address: "10.7.16.10",
      name: "Site-B-Indoor-DAS",
      port: "8080",
      web_username: "admin",
      web_password: System.get_env("BS_SITE_B_PASSWORD") || "passwo
    },

    # Site C - Rooftop
    %{
      address: "192.168.100.50",
      name: "Site-C-Rooftop",
      port: "8080",
      web_username: "admin",
      web_password: System.get_env("BS_SITE_C_PASSWORD") || "passwo
    },

    # Lab - Test Equipment
    %{
      address: "10.5.198.100",
      name: "Lab-Test-Airscale-01",
      port: "8080",
      web_username: "Nemuadmin",
      web_password: "nemuuser"
    },

    # Lab - Development
    %{
      address: "10.5.198.101",
      name: "Lab-Dev-Airscale-02",
      port: "8080",
      web_username: "Nemuadmin",
      web_password: "nemuuser"
```

```
      }
    ]
  }

#
# InfluxDB Configuration
#

config :ran_monitor, RanMonitor.InfluxDbConnection,
  auth: [
    username: System.get_env("INFLUX_USERNAME") || "monitor",
    password: System.get_env("INFLUX_PASSWORD") || "change_this_passw
  ],
  database: "nokia-monitor",
  host: System.get_env("INFLUX_HOST") || "10.179.2.135"
```

Related Documentation

- Operations Guide - Day-to-day operations
- AirScale Configuration Guide - Configuring base stations
- Nokia Counter Reference - Performance counter definitions
- Grafana Integration - Building dashboards and alerts
- API Endpoints - REST API reference
- Data Retention Policy - Managing data lifecycle

MDT Data Collection with TCE

Trace Collection Entity (TCE)

RAN Monitor includes an integrated Trace Collection Entity for capturing and analyzing LTE/5G protocol messages. This enables detailed troubleshooting, drive testing, and RF optimization.

What is TCE?

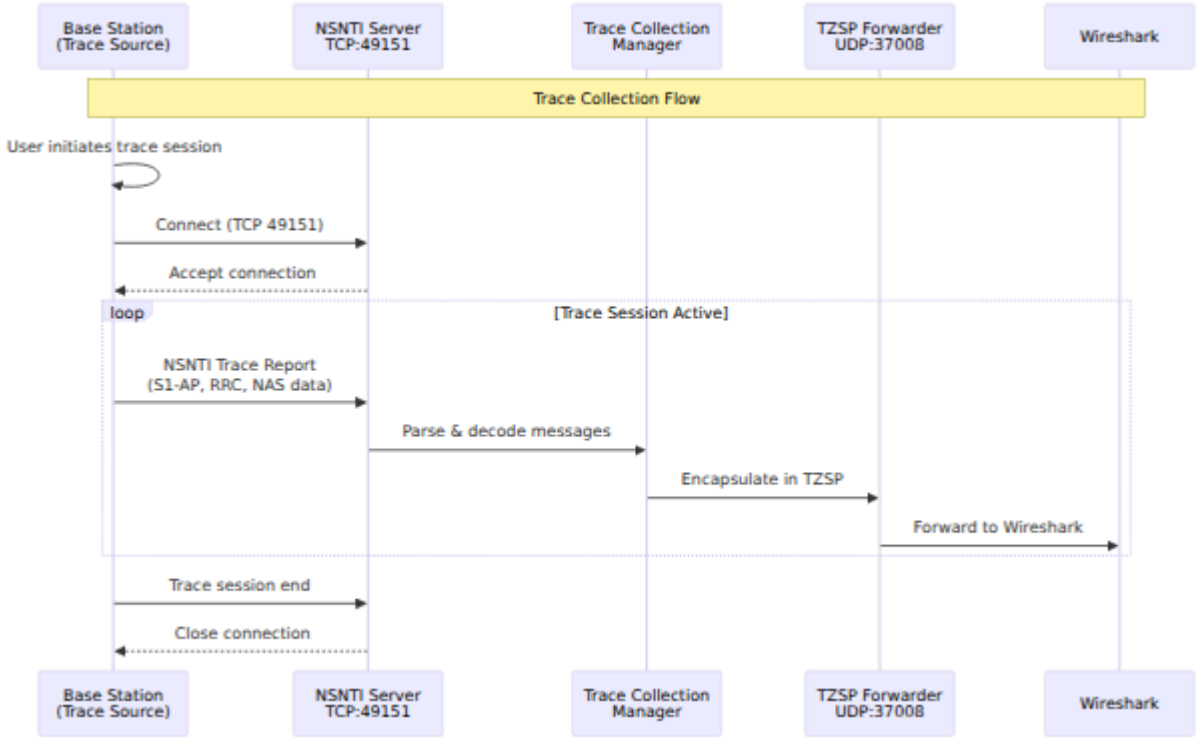
The Trace Collection Entity receives trace data from Nokia AirScale base stations containing:

- **S1-AP Messages** - Control plane signaling between eNodeB and EPC
- **RRC Messages** - Radio Resource Control signaling
- **NAS Messages** - Non-Access Stratum signaling
- **User Plane Data** - PDCP layer throughput information

TCE Components

Component	Protocol	Port	Purpose
NSNTI Server	TCP	49151	Receives trace messages from base stations
TZSP Server	UDP	37008	Forwards traces to Wireshark for real-time analysis
Protocol Decoders	ASN.1	-	Decodes S1-AP and RRC messages

How It Works



The Trace Collection page shows active connections, NSNTI listen port (49151), TZSP configuration, and connected base stations.

Trace Collection Setup

1. **Verify TCE is Running:**

```
ss -tlnp | grep 49151
# Should show: LISTEN 0.0.0.0:49151
```

2. **Configure Base Station Trace:**

   - Set trace destination IP to RAN Monitor server
   - Set trace destination port to 49151
   - Enable trace categories (S1-AP, RRC, NAS as needed)
   - Start trace session

3. **Configure Wireshark:**

   Basic Setup:

- Start capture on the interface receiving TZSP packets
- Use capture filter: `udp port 37008`

Protocol Decoding Configuration:

RAN Monitor uses specific UDP ports to identify different protocol types and RRC channels. Configure Wireshark's "Decode As" feature to properly decode these protocols:

Method 1: Using Wireshark GUI

i. Go to **Analyze → Decode As...**
ii. Click the **+** button to add new entries
iii. Configure each row as follows:

Field	Value	Type	Current	Decode As
udp.port	36412	Integer	(none)	SCTP
sctp.port	36412	Integer	(none)	S1AP
udp.port	37000	Integer	(none)	TZSP
udp.port	37001	Integer	(none)	LTE RRC (DL-CCCH)
udp.port	37002	Integer	(none)	LTE RRC (DL-DCCH)
udp.port	37003	Integer	(none)	LTE RRC (BCCH)
udp.port	37004	Integer	(none)	LTE RRC (PCCH)
udp.port	37008	Integer	(none)	TZSP
udp.port	37011	Integer	(none)	LTE RRC (UL-CCCH)
udp.port	37012	Integer	(none)	LTE RRC (UL-DCCH)
udp.port	38000	Integer	(none)	MAC-LTE
udp.port	38001	Integer	(none)	MAC-LTE (DL)
udp.port	38002	Integer	(none)	MAC-LTE (BCH)
udp.port	38003	Integer	(none)	MAC-LTE (PCH)
udp.port	38011	Integer	(none)	MAC-LTE (UL)
udp.port	38012	Integer	(none)	MAC-LTE (RACH)

Method 2: Using decode_as_entries file

Create or edit `~/.config/wireshark/decode_as_entries` (Linux/Mac) or `%APPDATA%\Wireshark\decode_as_entries` (Windows):

```
# RAN Monitor TZSP Port Mappings
decode_as_entry: udp.port,36412,(none),SCTP
decode_as_entry: sctp.port,36412,(none),S1AP
decode_as_entry: udp.port,37000,(none),TZSP
decode_as_entry: udp.port,37001,(none),LTE RRC
decode_as_entry: udp.port,37002,(none),LTE RRC
decode_as_entry: udp.port,37003,(none),LTE RRC
decode_as_entry: udp.port,37004,(none),LTE RRC
decode_as_entry: udp.port,37008,(none),TZSP
decode_as_entry: udp.port,37011,(none),LTE RRC
decode_as_entry: udp.port,37012,(none),LTE RRC
decode_as_entry: udp.port,38000,(none),MAC-LTE
decode_as_entry: udp.port,38001,(none),MAC-LTE
decode_as_entry: udp.port,38002,(none),MAC-LTE
decode_as_entry: udp.port,38003,(none),MAC-LTE
decode_as_entry: udp.port,38011,(none),MAC-LTE
decode_as_entry: udp.port,38012,(none),MAC-LTE
```

Port Reference Guide:

Port	Protocol	Channel/Type	Description
36412	S1AP	-	Standard S1AP control plane (eNodeB ↔ EPC)
37000	RRC	Generic	Fallback for unknown RRC channel types
37001	RRC	DL-CCCH	Downlink Common Control Channel
37002	RRC	DL-DCCH	Downlink Dedicated Control Channel
37003	RRC	BCCH-DL-SCH	Broadcast Control Channel (System Information)
37004	RRC	PCCH	Paging Control Channel
37008	TZSP	-	Main TZSP listener port
37011	RRC	UL-CCCH	Uplink Common Control Channel (RRC Connection Request)
37012	RRC	UL-DCCH	Uplink Dedicated Control Channel (Measurement Reports)
38000	MAC-LTE	Generic	Fallback for unknown MAC channel types
38001	MAC-LTE	Downlink	Downlink Shared Channel
38002	MAC-LTE	BCH	Broadcast Channel
38003	MAC-LTE	PCH	Paging Channel

Port	Protocol	Channel/Type	Description
38011	MAC-LTE	Uplink	Uplink Shared Channel
38012	MAC-LTE	RACH	Random Access Channel

Useful Display Filters:

```
# Show all TZSP packets
tzsp

# Show specific protocols
s1ap || rrc || mac-lte

# Show only uplink RRC messages
udp.port == 37011 || udp.port == 37012

# Show only downlink RRC messages
udp.port == 37001 || udp.port == 37002

# Show RRC connection establishment
rrc.rrcConnectionRequest || rrc.rrcConnectionSetup

# Show handover messages
s1ap.HandoverRequired || s1ap.HandoverCommand
```

Use Cases

Drive Testing:

- Capture end-user RF experience
- Analyze handover performance
- Measure signal quality (RSRP, RSRQ, SINR)
- Identify coverage holes

Troubleshooting:

- Debug call setup failures

- Analyze handover issues
- Investigate dropped calls
- Review mobility events

RF Optimization:

- PCI planning validation
- Neighbor relation optimization
- Handover parameter tuning
- Coverage and capacity analysis

Overview

Minimization of Drive Tests (MDT) allows you to collect radio measurements (RSRP, RSRQ, coverage data) directly from UEs without traditional drive testing. This guide shows you how to capture MDT data from Nokia AirScale base stations using the Omnitouch RAN Monitor Web UI and view it in Wireshark.

Architecture

The TCE (Trace Collection Entity) is integrated into the Omnitouch RAN Monitor and handles the conversion of Nokia-specific Ne3s protocol traces into standard formats viewable in Wireshark.

Prerequisites

Required Licenses

The Nokia Airscale requires feature activations including **Per-Call Measurement Data** to collect this data, and these features enabled and configured,

Contact ONS if you need help with licensing or have questions about your specific deployment.

System Requirements

- Omnitouch RAN Monitor with TCE running
- Wireshark 3.0+ installed on your machine
- TCE Lua plugins installed in Wireshark (see TCE README)
- Network connectivity to the AirScale

Setting Up MDT Tracing

The TCE built into the RAN Monitor converts the incoming Nokia data into standard Wireshark viewable formats.

Step 1: Configure the Trace Collection Entity

Use the RAN Monitor Web UI to set up the base station to send traces to the TCE:

1. Open Web UI: `https://<ran-monitor-ip>:9443`
2. Navigate to **Base Stations** page
3. Click on the device you want to trace
4. Go to **Configuration Management** section
5. Download the current configuration (backup)
6. Edit the configuration to add/update TCE settings:
   - **Trace Collection Entity IP**: `<Your RAN Monitor IP>`
   - **Trace Collection Entity Port**: `49151`
7. Upload the modified configuration
8. Validate the configuration (wait for validation to complete)
9. Activate the configuration

For help with specific configuration parameters or AirScale software versions, contact ONS.

Step 2: Configure MDT on the AirScale

Enable MDT tracing on your base station. Configuration options include:

- **Trace Type**: Immediate MDT (real-time) or Logged MDT (idle mode)
- **Area Scope**: Cell-specific, Tracking Area, or PLMN-wide
- **Measurement Interval**: How often UEs report (e.g., 5000ms)
- **Measurement Type**: RSRP, RSRQ, or both
- **Trace Depth**: Minimum, Medium, or Maximum

Contact ONS for guidance on configuring these parameters for your specific use case.

Step 3: Activate the Trace Session

Once configured, activate the trace session on the AirScale. The base station will begin sending MDT data to the TCE which in turn will forward it to your monitoring machine.

Viewing MDT Data in Wireshark

Setup Wireshark Capture

1. Start Wireshark on your machine
2. Capture on the **loopback interface** (`lo` on Linux, `lo0` on macOS, `Loopback` on Windows)
3. Set capture filter: `udp port 37008`
4. Start capturing

Example Wireshark capture showing S1AP control plane messages (InitialUEMessage, Attach request), LTE RRC messages (RRCConnectionReject, RRCConnectionReestablishment), and various signaling flows captured via the TCE.

Filter for MDT Measurements

Once data is flowing, use these display filters:

```
# Show all RRC Measurement Reports
lte-rrc.measurementReport

# Show all uplink RRC messages
udp.dstport >= 37011 && udp.dstport <= 37012

# Filter by poor RSRP (< -100 dBm)
lte-rrc.rsrpResult < 40

# Filter by poor RSRQ (< -12 dB)
lte-rrc.rsrqResult < 22
```

Understanding the Data

MDT measurements appear as **RRC MeasurementReport** messages containing:

- **Serving Cell Measurements**: RSRP and RSRQ for the connected cell
- **Neighbor Cell Measurements**: RSRP and RSRQ for nearby cells
- **Cell IDs**: Physical cell IDs for correlation

- **GPS Location**: If configured and supported by UE

Expand the RRC message in Wireshark to see detailed measurements:

```
Radio Resource Control (RRC)
└─ UL-DCCH-Message
    └─ message: measurementReport
        └─ MeasurementReport
            └─ measResults
                ├─ measResultServCell (serving cell RSRP/RSRQ)
                └─ measResultNeighCells (neighbor cell
measurements)
```

Export for Analysis

To analyze data offline:

1. **File → Export Packet Dissections → As CSV**
2. Include fields: `lte-rrc.rsrpResult`, `lte-rrc.rsrqResult`, `lte-rrc.physCellId`
3. Process in Excel, Python, or other tools

Common Use Cases

Coverage Analysis: Look for areas with weak RSRP/RSRQ

```
lte-rrc.rsrpResult < 40 || lte-rrc.rsrqResult < 22
```

Handover Analysis: See which neighbor cells UEs are reporting

```
lte-rrc.MeasResultListEUTRA
```

Interference Detection: Good RSRP but poor RSRQ indicates interference

```
lte-rrc.rsrpResult > 50 && lte-rrc.rsrqResult < 20
```

Troubleshooting

No data in Wireshark?

- Verify TCE is running: `ps aux | grep beam`
- Check Wireshark is capturing loopback with filter `udp port 37008`
- Confirm trace session is active on AirScale
- Verify TCE IP/port configured correctly on base station

Incomplete data?

- Check licenses are active (MDT + Per-Call Measurement)
- Increase trace depth to MAXIMUM
- Ensure UEs support MDT (LTE Release 10+)

For configuration help, licensing issues, or AirScale-specific questions, contact ONS.

Quick Start Checklist

- [ ] Verify MDT and per-call measurement licenses are active
- [ ] Configure TCE IP (RAN Monitor IP) and port 49151 on AirScale
- [ ] Start TCE on RAN Monitor server
- [ ] Activate MDT trace session on base station
- [ ] Start Wireshark capture on loopback with filter `udp port 37008`
- [ ] Apply display filter: `lte-rrc.measurementReport`
- [ ] Analyze measurements and export as needed

Support

- **Omnitouch Network Services (ONS)**: For AirScale configuration, licensing, and deployment assistance

Troubleshooting Guide

Problem Resolution for RAN Monitor

Common issues, diagnostic procedures, and solutions

Table of Contents

Overview

This guide helps you diagnose and resolve common issues with RAN Monitor. Each section provides symptoms, diagnosis steps, and solutions.

Troubleshooting Approach

1. Identify the Symptom

- What is not working as expected?
- When did the problem start?
- What changed recently?

2. Gather Information

- Check application logs
- Review device status in Web UI
- Check database connectivity
- Review recent configuration changes

3. Diagnose the Root Cause

- Use diagnostic tools
- Review error messages
- Test individual components
- Isolate the problem

4. Implement Solution

- Apply fix based on diagnosis
- Verify solution resolves issue
- Monitor for recurrence
- Document findings

Before You Start

Check the Basics:

- Is RAN Monitor running? (`ps aux | grep ran_monitor`)
- Are required services running? (MySQL, InfluxDB)
- Is network connectivity working?
- Have there been recent changes?

Device Connection Issues

Problem: Device Not Registering

Symptoms:

- Device shows "Not Registered" in Web UI
- Red (failed) status in Base Stations page
- No metrics are being collected from device
- Error messages in application logs

Diagnostic Steps:

1. Verify Network Connectivity

```
# Test basic connectivity
ping <device-ip>

# Test management port
telnet <device-ip> 8080
```

Expected: Successful ping and telnet connection **If Fails:** Network issue - check routes, firewall, device status

2. Check Configuration

In Web UI → Base Stations → Click device → Review configuration:

- Is IP address correct?
- Is port correct (typically 8080)?
- Are credentials configured?

In `config/runtime.exs`:

```
%{
  address: "10.7.15.66",  # Correct IP?
  name: "Site-A-BS1",
  port: "8080",           # Correct port?
  web_username: "admin",  # Correct username?
  web_password: "password" # Correct password?
}
```

3. Check Application Logs

Web UI → Application Logs → Filter for device name

Look for:

- `[error] Authentication failed` → Incorrect credentials
- `[error] Connection refused` → Port/firewall issue
- `[error] Timeout` → Network connectivity problem
- `[error] Certificate error` → Manager key/certificate issue

Solutions:

Network Issue:

1. Verify device is powered on and operational
2. Check network routes between RAN Monitor and device
3. Verify firewall allows:
    - RAN Monitor → Device port 8080
    - Device → RAN Monitor port 9076 (webhooks)
4. Test from RAN Monitor server directly

Incorrect Credentials:

1. Verify credentials work directly on device WebLM interface
2. Update credentials in `config/runtime.exs`
3. Restart RAN Monitor
4. Monitor logs for successful registration

Port/Firewall Issue:

1. Verify correct port in configuration
2. Check firewall rules on both sides
3. Test port accessibility: `telnet <device-ip> 8080`
4. Review device-side security settings

Manager Key/Certificate Issue:

1. Verify files exist:
    - `priv/external/nokia/ne.key.pem`

    - `priv/external/nokia/ne.cert.der`
2. Check file permissions (should be readable)
3. Verify files are valid Nokia manager credentials
4. Contact Nokia support if keys are invalid

Problem: Session Keeps Expiring

Symptoms:

- Device repeatedly disconnects and reconnects
- "Session expired" messages in logs
- Intermittent red/green status in Web UI
- Gaps in metric collection

Diagnostic Steps:

1. Check Session Information

Web UI → Base Stations → Click device → Session Lifecycle:

- What is session expiry time?
- Is keep-alive working?
- How often is session expiring?

2. Check Keep-Alive Interval

In `config/runtime.exs`:

```
nokia: %{
  ne3s: %{
    reregister_interval: 30  # Should be 30-60 seconds
  }
}
```

3. Check Network Stability

- Are there intermittent network issues?

- Check for packet loss: `ping <device-ip> -c 100`
- Review network logs for flapping interfaces

4. Check Clock Synchronization

```
# On RAN Monitor server
date

# On device (if accessible)
# Verify time is synchronized
```

Solutions:

Keep-Alive Interval Too Long:

1. Reduce `reregister_interval` to 30 seconds
2. Restart RAN Monitor
3. Monitor session stability

Network Instability:

1. Work with network team to diagnose
2. Check for intermittent connectivity
3. Review switch/router logs
4. Consider redundant network paths

Clock Synchronization:

1. Configure NTP on both RAN Monitor and devices
2. Verify clocks are synchronized
3. Check for large time differences

Device-Side Issue:

1. Check device logs for errors
2. Verify device management interface is stable
3. Consider device reboot if software issue suspected

Problem: Metrics Not Appearing

Symptoms:

- Device shows as "Associated" (green) in Web UI
- But no metrics appearing in InfluxDB
- No data in Grafana dashboards
- InfluxDB Status page shows zero or low counts

Diagnostic Steps:

1. Verify Device is Associated

Web UI → Base Stations:

- Device status is green?
- Last contact timestamp is recent?
- Session is active?

2. Check InfluxDB Connection

Web UI → InfluxDB Status:

- Connection status green?
- Can RAN Monitor write to InfluxDB?

Test connectivity:

```
# From RAN Monitor server
curl http://<influxdb-host>:8086/ping
```

3. Check Application Logs

Look for:

- `[error] InfluxDB write failed` → Connection or permission issue
- `[error] Failed to collect metrics` → Device communication issue
- `[info] Metrics collected: 0` → Device not returning data

4. Check InfluxDB Directly

Query InfluxDB for recent data:

```
# InfluxDB 1.x
influx -database 'nokia-monitor' -execute '
  SELECT COUNT(*) FROM PerformanceMetrics
  WHERE basebandName='''Site-A-BS1'''
  AND time > now() - 1h
'

# InfluxDB 2.x
influx query 'from(bucket:"nokia-monitor")
  |> range(start: -1h)
  |> filter(fn: (r) => r.basebandName == "Site-A-BS1")
  |> filter(fn: (r) => r._measurement == "PerformanceMetrics")
  |> count()'
```

Solutions:

InfluxDB Connection Issue:

1. Verify InfluxDB is running
2. Check `config/runtime.exs` for correct:
   - Host address
   - Port (8086)
   - Database/bucket name
   - Credentials/API token
3. Test connectivity from RAN Monitor server
4. Verify firewall allows port 8086
5. Restart RAN Monitor after fixing configuration

InfluxDB Permission Issue:

1. Verify credentials have write permission to bucket/database
2. Check InfluxDB logs for authentication errors
3. Recreate API token with proper permissions
4. Update `config/runtime.exs` with new token

5. Restart RAN Monitor

InfluxDB Storage Full:

1. Check disk space: `df -h`
2. Review retention policies
3. Clean old data or expand storage
4. See Data Retention Policy Guide

Device Not Returning Data:

1. Check device is configured to send metrics
2. Verify webhook URL is correct in device configuration
3. Check device logs for errors
4. Verify RAN Monitor webhook receiver is running (port 9076)

Data Collection Problems

Problem: Gaps in Historical Data

Symptoms:

- Grafana dashboards show gaps in time series
- Missing data points for certain time periods
- InfluxDB queries return incomplete results

Diagnostic Steps:

1. Check Application Uptime

Were there service interruptions during gap period?

```
# Check system logs for restarts
journalctl -u ran_monitor --since "2025-12-29" --until "2025-12-
30"
```

2. Check Device Connectivity History

Web UI → Base Stations → Device → Review "Last Contact" history

- Was device connected during gap period?
- Are there connectivity issues?

3. Check InfluxDB Availability

Were there InfluxDB outages during gap period?

- Check InfluxDB logs
- Review monitoring/alerting history

Solutions:

RAN Monitor Downtime:

- Data gap is normal during service interruption
- Historical data cannot be backfilled
- Document incident and restore service

Device Disconnection:

- Investigate why device disconnected
- Fix connectivity issue
- Data gap normal during disconnection
- Future data will resume collection

InfluxDB Outage:

- Metrics were likely collected but not stored
- Check RAN Monitor logs for write failures
- Restore InfluxDB service
- Data gap cannot be recovered

Prevention:

- Implement monitoring for RAN Monitor uptime

- Set up alerts for extended disconnections
- Monitor InfluxDB health
- Consider HA/redundancy for critical systems

Web UI Issues

Problem: Cannot Access Web UI

Symptoms:

- Browser cannot connect to https://<ran-monitor-ip>:9443
- Connection timeout or refused
- SSL certificate errors

Diagnostic Steps:

1. Verify Web UI is Running

Check application logs:

```
[info] Running ControlPanelWeb.Endpoint with cowboy
```

Check process:

```
ps aux | grep control_panel
netstat -tulpn | grep 9443
```

2. Test Connectivity

From another machine:

```
telnet <ran-monitor-ip> 9443
```

From RAN Monitor server itself:

```
curl -k https://localhost:9443
```

3. Check Firewall

```
# Check if port is open
sudo iptables -L -n | grep 9443

# Or
sudo firewall-cmd --list-ports
```

Solutions:

Port Not Open:

1. Add firewall rule:
   ```
   sudo firewall-cmd --add-port=9443/tcp --permanent
   sudo firewall-cmd --reload
   ```
2. Test access again

Web UI Not Started:

1. Check `config/runtime.exs` for web endpoint configuration
2. Verify SSL certificate files exist
3. Check application logs for startup errors
4. Restart RAN Monitor

SSL Certificate Issues:

1. Verify certificate files exist and are readable:
   ```
   ls -l priv/cert/omnitouch.pem
   ls -l priv/cert/omnitouch.crt
   ```
2. Check certificate validity:
   ```
   openssl x509 -in priv/cert/omnitouch.crt -text -noout
   ```

3. Regenerate if expired or missing
4. Restart RAN Monitor

Wrong Port:

1. Check `config/runtime.exs` for configured port
2. Use correct port in browser
3. Or set `CONTROL_PANEL_HTTPS_PORT` environment variable

Problem: Web UI Loads But Shows No Data

Symptoms:

- Web UI is accessible
- Pages load but show empty lists or zero counts
- Dashboard shows no devices

Diagnostic Steps:

1. Check Device Configuration

Is anything configured in `config/runtime.exs`?

```
airscales: [
  # Should have at least one device
]
```

2. Check Database Connection

Are devices stored in MySQL?

```
mysql -u ran_monitor_user -p ran_monitor -e "SELECT * FROM
airscales;"
```

3. Check Application Logs

Look for database connection errors or query failures.

Solutions:

No Devices Configured:

1. Add devices to `config/runtime.exs`
2. Restart RAN Monitor
3. Devices should appear in Web UI

Database Connection Issue:

1. Verify MySQL is running
2. Check connection configuration in `config/runtime.exs`
3. Test database connection
4. Restart RAN Monitor

Database Problems

Problem: MySQL Connection Errors

Symptoms:

- Application logs show database connection errors
- Web UI shows errors loading pages
- "Database connection timeout" messages

Diagnostic Steps:

1. Verify MySQL is Running

```
systemctl status mysql
# or
systemctl status mariadb
```

2. Test Connection

From RAN Monitor server:

```
mysql -h <mysql-host> -u <username> -p <database>
```

3. Check Configuration

In `config/runtime.exs`:

```
config :ran_monitor, RanMonitor.Repo,
  username: "ran_monitor_user",
  password: "password",
  hostname: "localhost",
  database: "ran_monitor",
  pool_size: 10
```

Solutions:

MySQL Not Running:

1. Start MySQL service:
   ```
   systemctl start mysql
   ```
2. Verify it starts correctly
3. RAN Monitor will reconnect automatically

Connection Configuration Error:

1. Verify hostname, username, password, database name
2. Test connection manually
3. Update `config/runtime.exs` if incorrect
4. Restart RAN Monitor

Network Issue:

1. Check network connectivity to MySQL server
2. Verify firewall allows port 3306
3. Check MySQL bind address (should allow remote connections if needed)

Too Many Connections:

1. Check MySQL max_connections setting
2. Reduce pool_size in configuration if needed
3. Restart RAN Monitor

Performance Issues

Problem: High CPU or Memory Usage

Symptoms:

- RAN Monitor using excessive CPU or RAM
- System becomes slow or unresponsive
- Database connections timing out
- Response time degraded

Diagnostic Steps:

1. Check Resource Usage

```
# CPU and memory
top -p $(pgrep -f ran_monitor)

# Detailed process info
ps aux | grep ran_monitor
```

2. Check Number of Monitored Devices

How many devices are configured?

- More devices = more resources needed
- Check if device count recently increased

3. Check Collection Intervals

Are polling intervals very frequent?

- More frequent = higher CPU/network usage
- Default is 10 seconds for metrics

4. Check Database Pool Size

In `config/runtime.exs`:

```
pool_size: 10  # May need adjustment
```

Solutions:

Too Many Devices for Resources:

1. Monitor resource usage trends
2. Increase server resources (CPU/RAM)
3. Or reduce number of monitored devices
4. Consider scaling horizontally (multiple instances)

Database Pool Too Large:

1. Reduce pool_size in configuration
2. Rule of thumb: 2 connections per device + 5 for Web UI
3. Restart RAN Monitor
4. Monitor resource usage

Memory Leak:

1. Monitor memory usage over time
2. If continuously increasing, may be memory leak
3. Restart RAN Monitor as temporary fix
4. Report issue with logs and metrics

InfluxDB Write Performance:

1. Check InfluxDB resource usage
2. Verify InfluxDB isn't bottleneck
3. Consider separate InfluxDB server
4. Review retention policies to reduce data volume

Problem: Slow Web UI Response

Symptoms:

- Web UI takes long time to load pages
- Dashboard is sluggish
- Timeouts when viewing device details

Diagnostic Steps:

1. Check Server Resources

Is RAN Monitor server overloaded?

```
top
free -h
df -h
```

2. Check Database Performance

Are database queries slow?

```
# MySQL slow query log
mysql -u root -p -e "SHOW VARIABLES LIKE 'slow_query_log%';"
```

3. Check Network Latency

Is there high latency to database or clients?

Solutions:

Server Resource Issue:

1. Reduce load on server
2. Increase server resources
3. Move databases to separate servers

Database Performance:

1. Optimize MySQL configuration
2. Add indexes if needed (tables should have them)
3. Increase database server resources

Network Latency:

1. Investigate network path
2. Consider moving components closer
3. Use local database if possible

Alarm Problems

Problem: Alarms Not Appearing

Symptoms:

- Known faults not showing in Alarms page
- Alarm count is zero when faults exist
- Delayed alarm notifications

Diagnostic Steps:

1. Check Device is Sending Alarms

Verify in device management interface that alarms are configured to be sent.

2. Check Webhook Receiver

Is webhook endpoint running?

```
netstat -tulpn | grep 9076
```

Look for:

```
tcp  0  0.0.0.0:9076  0.0.0.0:*  LISTEN
```

3. Check Webhook Configuration

In device configuration, verify webhook URL points to RAN Monitor:

```
http://<ran-monitor-ip>:9076/webhook
```

4. Check Application Logs

Look for webhook receiver errors or alarm parsing failures.

5. Check InfluxDB

Are alarms being written?

```
influx -database 'nokia-monitor' -execute '
  SELECT COUNT(*) FROM Alarms WHERE time > now() - 1h
'
```

Solutions:

Webhook Receiver Not Running:

1. Check `config/runtime.exs` for webhook endpoint configuration
2. Verify port 9076 is configured
3. Restart RAN Monitor
4. Verify port is listening

Device Not Sending:

1. Configure device to send alarm notifications
2. Verify webhook URL in device configuration
3. Test alarm generation on device

Firewall Blocking:

1. Verify device can reach RAN Monitor port 9076
2. Add firewall rule if needed
3. Test connectivity: `telnet <ran-monitor-ip> 9076` from device network

InfluxDB Write Failure:

1. Check InfluxDB connection
2. Verify write permissions
3. Check InfluxDB storage capacity
4. Review application logs for write errors

Diagnostic Tools

Application Logs

Access via Web UI:

1. Navigate to Application Logs page
2. Filter by log level
3. Search for keywords
4. Pause to review specific errors

Access via Command Line:

If running as systemd service:

```
journalctl -u ran_monitor -f
```

If running via mix:

- Logs appear in console output

Log Levels:

- Emergency/Alert/Critical - System-critical issues

- Error - Errors that need attention
- Warning - Potential issues
- Info - Normal operational messages
- Debug - Detailed diagnostic information

Useful Search Terms:

- Device name (e.g., "Site-A-BS1")
- "error" or "failed"
- "InfluxDB" or "MySQL"
- "registration" or "session"

InfluxDB Queries

Query for recent metrics:

```
influx -database 'nokia-monitor' -execute '
  SELECT * FROM PerformanceMetrics
  WHERE basebandName='''Site-A-BS1'''
  AND time > now() - 5m
  LIMIT 10
'
```

Count metrics by device:

```
influx -database 'nokia-monitor' -execute '
  SELECT COUNT(*) FROM PerformanceMetrics
  GROUP BY basebandName
'
```

Query for alarms:

```
influx -database 'nokia-monitor' -execute '
  SELECT * FROM Alarms
  WHERE time > now() - 1h
'
```

MySQL Queries

Check configured devices:

```
SELECT name, address, port, registration_status
FROM airscales;
```

Check for errors in database:

```
mysql -u ran_monitor_user -p ran_monitor -e "SHOW PROCESSLIST;"
```

Network Diagnostics

Test connectivity:

```
# Basic connectivity
ping <device-ip>

# Port accessibility
telnet <device-ip> 8080
nc -zv <device-ip> 8080

# Trace route
traceroute <device-ip>
```

Check firewall:

```
# List rules
sudo iptables -L -n -v

# Check specific port
sudo iptables -L -n | grep 8080
```

Getting Help

Before Contacting Support

Gather the following information:

1. Problem Description

- What is not working?
- When did it start?
- What changed recently?

2. Error Messages

- Copy exact error messages from logs
- Include timestamps
- Note frequency of errors

3. System Information

- RAN Monitor version
- Operating system and version
- Database versions (MySQL, InfluxDB)
- Number of monitored devices

4. Diagnostic Results

- Results from diagnostic steps above
- Relevant log excerpts
- Configuration (sanitize passwords)

5. Impact

- How many devices affected?
- Is this blocking operations?
- What is the business impact?

Documentation Resources

- **Web UI Guide** - Control panel reference
- **Common Operations Guide** - Routine tasks
- **Runtime Configuration Guide** - Configuration details
- **Grafana Integration Guide** - Analytics setup
- **Alarm Management Guide** - Alarm handling
- **Data Retention Policy Guide** - Data management
- **Operations Guide** - Complete overview

Self-Service Resources

Check Logs First:

- Application Logs page in Web UI
- System logs: `journalctl -u ran_monitor`
- Database logs

Review Recent Changes:

- Configuration file modifications
- Device additions/removals
- Network changes
- Software updates

Test Basic Functionality:

- Can you access Web UI?
- Are devices showing as connected?
- Is InfluxDB accessible?
- Are metrics flowing?

Escalation

If you cannot resolve the issue:

1. Document all diagnostic steps taken
2. Gather information listed above
3. Contact Omnitouch support with details
4. Be prepared to provide:
    - Configuration files (sanitized)
    - Log excerpts
    - Screenshots if relevant
    - Steps to reproduce

Related Documentation

- **Operations Guide** - Complete operational overview
- **Web UI Guide** - Control panel user guide
- **Common Operations Guide** - Day-to-day tasks
- **Alarm Management Guide** - Alarm handling procedures
- **Runtime Configuration Guide** - Configuration reference
- **Grafana Integration Guide** - Analytics and dashboards
- **Data Retention Policy Guide** - Data lifecycle management

Web UI Guide

RAN Monitor Control Panel - User Interface Reference

Complete guide to using the RAN Monitor web-based control panel

---

Table of Contents

Overview

RAN Monitor includes a built-in web-based control panel for real-time operational monitoring and management. The Web UI provides immediate visibility into device status, alarms, configuration, and system health.

Web UI vs. Grafana

Web UI is Best For:

- Immediate device status checks
- Real-time alarm monitoring
- Configuration management
- Session troubleshooting
- System administration

Grafana is Best For:

- Historical trend analysis
- Custom KPI dashboards
- Long-term capacity planning
- Pattern identification
- Executive reporting

For Grafana dashboards and analytics, see the Grafana Integration Guide.

Accessing the Web UI

The control panel is accessed via HTTPS:

URL: `https://<ran-monitor-ip>:9443`

Default Port: 9443 (configurable via `CONTROL_PANEL_HTTPS_PORT` environment variable)

SSL Certificates:

- Self-signed certificates work for lab environments
- Production should use CA-signed certificates
- Certificates configured in `config/runtime.exs`

For configuration details, see the Runtime Configuration Guide.

Auto-Refresh: Most pages automatically refresh every 5 seconds to show real-time data.

Main Dashboard

The dashboard provides an at-a-glance view of your RAN infrastructure.

Key Sections

System Status

- Overall health indicators
- System uptime and connectivity

Device Summary

- Count of associated/failed devices
- Registration status overview
- Quick device health snapshot

Active Alarms

- Current fault count by severity
- Color-coded severity levels (Critical, Major, Minor, Warning)
- Quick links to alarm details

Recent Activity

- Latest events and changes
- Configuration updates
- Session status changes

Features

- Auto-refreshes every 5 seconds
- Color-coded status indicators (green = healthy, red = issues)

- Click-through navigation to detailed views
- Real-time metric updates

Base Stations Page

View all managed devices with their current status and session information.

URL: `https://<ran-monitor-ip>:9443/nokia/enodeb`

The NOKIA eNodeB Status page showing device list with connection status, session state, and action buttons.

Statistics Summary

The top bar shows aggregate device counts:

Statistic	Description
Total Devices	Number of configured devices
Connected	Devices with active sessions
Pending	Devices awaiting registration
Disconnected	Devices with no active session

Device Table

Column	Description
Name	Device name as configured
Status	Connection status: "Connected" (green) or "Disconnected" (red)
Address	Device IP address and port
Session	Session state: "Active" (green) or "Inactive" (gray)
Actions	Device action buttons

Action Buttons

Each device row has action buttons:

Button	Description
Ping	Test network connectivity to the device
Config	View current device configuration
Config Ops	Access configuration management operations (download, upload, validate, activate)
Force Retry	Force re-registration attempt for disconnected devices

Device Details Panel

Clicking a device row shows additional details:

Field	Description
Manager ID	Internal manager identifier
Session ID	Current session identifier
Agent Type	Device agent type (e.g., COMA)
Vendor	Device vendor (Nokia)

Filtering and Search

- Filter by connection status
- Search by device name or IP address
- Sort by any column

Device Detail View

Click any device from the Base Stations page to see comprehensive information.

Registration Details

- Manager identity and authentication status
- Registration timestamp
- Authentication credentials in use
- Manager keys and certificates

Session Lifecycle

- Session creation time
- Session expiry time
- Keep-alive interval and status
- Last keep-alive timestamp
- Time remaining until expiry

Recent Metrics

- Latest performance data snapshots
- Counter values and timestamps
- Metric collection status
- Data collection intervals

Active Alarms

- Current faults for this specific device
- Alarm severity and description
- Alarm timestamps
- Probable cause information

Configuration State

- Current parameter values
- Recent configuration changes
- Configuration timestamp
- Parameter change history

Alarms Page

Monitor all faults across your network in one centralized view.

Alarm Information

Severity Levels:

- **Critical** (Red) - Service affecting, immediate action required
- **Major** (Orange) - Significant degradation, urgent attention needed
- **Minor** (Yellow) - Non-service affecting, should be addressed
- **Warning** (Blue) - Informational, monitor for trends
- **Cleared** (Green) - Previously active alarm has been resolved

Alarm Details:

- Description of the issue
- Probable cause
- Affected system (DN - Distinguished Name)
- Timestamps (when alarm occurred and last updated)

Features

Color Coding:

- Immediate visual identification of severity
- Red = Critical alarms

- Orange = Major alarms
- Yellow = Minor alarms
- Blue = Warnings
- Green = Cleared

Sorting and Filtering:

- Sort by severity, device, or time
- Filter by alarm type
- Search for specific issues

Device Links:

- Click alarm to view affected device details
- Cross-reference with device metrics
- Navigate to device configuration

For detailed alarm handling procedures, see the Alarm Management Guide.

Configuration Management

The Web UI provides tools for managing device configurations safely and efficiently.

Download Configuration

Purpose: Retrieve and backup current configuration

Steps:

1. Navigate to device detail page
2. Click "Download Configuration"
3. Configuration is retrieved from device
4. Save configuration as XML file

Best Practice: Always download and save configuration before making changes

Upload Configuration

Purpose: Apply new configuration to device

Steps:

1. Select XML configuration file
2. Click "Upload Configuration"
3. Configuration is uploaded to device (creates a "plan")
4. System returns a Plan ID for tracking

Important: Upload only creates a plan - it does not activate the configuration

Validate Configuration

Purpose: Verify configuration is valid before activation

Steps:

1. Enter Plan ID from upload
2. Click "Validate"
3. Device validates syntax and parameters
4. System confirms readiness for activation or reports errors

Note: Always validate before activating to prevent configuration errors

Activate Configuration

Purpose: Apply validated configuration plan

Steps:

1. Enter validated Plan ID
2. Click "Activate Configuration"
3. Changes take effect immediately on device

4. Monitor status for success/failure

Warning: Activation is immediate and can affect service - ensure validation passed first

Configuration Workflow

Recommended Process:

1. Download current configuration (backup)
2. Modify configuration offline
3. Upload new configuration (get Plan ID)
4. Validate configuration (verify no errors)
5. Activate if validation succeeds
6. Verify changes took effect
7. Monitor device for stability

For base station configuration details, see the AirScale Configuration Guide.

Unconfigured eNodeBs Page

Discover and manage base stations attempting to connect that aren't yet configured in the system.

Purpose

The Unconfigured eNodeBs page helps you:

- Discover new base stations on the network
- Identify devices attempting unauthorized connections
- Verify device identifiers before adding to configuration
- Track connection attempts from unknown equipment

Information Displayed

Agent ID

- Device identifier detected from connection attempts
- Use this ID when adding device to configuration

Last Seen

- Most recent connection attempt timestamp
- Helps identify active vs. inactive devices

Occurrences

- Number of times the device attempted to connect
- Frequent attempts may indicate misconfiguration

First Seen

- When the device was first detected
- Useful for tracking new equipment

Actions Available

Refresh

- Reload the list of unconfigured devices
- Updates timestamps and occurrence counts

Delete

- Remove individual entries from the list
- Useful for cleaning up old/decommissioned devices

Clear All

- Remove all unconfigured device records
- Fresh start for the list

Configuration Help

When devices appear here, follow these steps:

1. **Note the Agent ID** from the table
2. **Add device configuration** to `config/runtime.exs`:

```
airscales: [
  %{
    address: "10.7.15.66",
    name: "Site-A-BS1",
    port: "8080",
    web_username: "admin",
    web_password: "password"
  }
]
```

3. **Restart RAN Monitor** to begin monitoring the device

For detailed configuration instructions, see the Runtime Configuration Guide.

Use Cases

- **Network Discovery:** Find new base stations added to network
- **Security:** Identify unauthorized connection attempts
- **Provisioning:** Verify device identifiers before configuration
- **Decommissioning:** Track attempts from devices that should be offline

---

Application Logs Page

Real-time logging dashboard for troubleshooting and monitoring system activity.

Log Levels

Filter by Log Level:

- **Emergency** - System-critical failures
- **Alert** - Immediate action required
- **Critical** - Critical conditions
- **Error** - Error conditions
- **Warning** - Warning conditions
- **Notice** - Normal but significant
- **Info** - Informational messages
- **Debug** - Detailed debugging information

Note: When filtering, the selected level and all higher severity levels are shown.

Features

Search and Filter:

- Text search across all log messages
- Real-time log streaming (last 500 messages)
- Filter by log level

Controls:

- **Pause/Resume** - Stop live log streaming to review messages
- **Clear** - Remove all logs from display
- **System Level** - Change application-wide log level dynamically

Color Coding:

- Red - Emergency/Alert/Critical levels
- Light Red - Error level
- Yellow - Warning level
- Cyan - Notice level
- Blue - Info level
- Gray - Debug level

Use Cases

Troubleshoot Connection Issues:

- Filter for errors from specific devices
- Search for device names or IP addresses
- Review connection failure messages

Monitor System Activity:

- Watch info-level logs for normal operations
- Track device registration events
- Monitor data collection activity

Debug Problems:

- Temporarily set debug level
- Reproduce the issue
- Review detailed logs
- Revert to info level when done

Investigate Failures:

- Search for error messages and stack traces
- Review timestamps around failure time
- Correlate with device events

Best Practices

- **Use Pause** when reviewing specific error sequences
- **Set appropriate log level:**
  - Info for production
  - Debug for troubleshooting
  - Warning for quiet production
- **Search effectively** using device names or error keywords
- **Log level changes persist** until application restart

Data Retention Policy Page

Manage how long data is stored in InfluxDB for each base station.

Global Settings Display

Default Retention Period

- System-wide retention policy in hours/days
- Configured in `config/config.exs`
- Default: 720 hours (30 days)

Total Records

- Count of all data points across all devices
- Updated on page refresh

Auto-Cleanup Status

- Shows cleanup runs hourly
- Background worker status

Per-Device Settings

For each configured base station:

Device Information:

- Device name
- Registration status (Registered/Not Registered)
- Current retention period setting

Record Counts:

- **Performance Metrics** - Number of PM data points stored
- **Configuration** - Number of configuration snapshots
- **Alarms** - Number of alarm records
- **Total** - Sum of all records for this device

Actions:

- **Update Retention Period** - Change retention hours (applies to this device only)
- **Clean Old Data** - Manually trigger cleanup based on retention period
- **Clear All Data** - Delete all data for this device (irreversible)

How Retention Works

1. **Global Default** - Set in configuration file, applies to all devices
2. **Per-Device Override** - Optionally set custom retention for specific devices
3. **Automatic Cleanup** - Runs hourly, deletes data older than retention period
4. **Manual Cleanup** - Use "Clean Old Data" to force immediate cleanup

Common Retention Periods

- **720 hours (30 days)** - Short-term operational monitoring
- **2160 hours (90 days)** - Standard retention for most deployments
- **4320 hours (180 days)** - Extended retention for compliance
- **8760 hours (365 days)** - Long-term historical analysis

Use Cases

- Reduce storage usage by lowering retention period
- Keep critical device data longer than others
- Clean up test data before production
- Manage InfluxDB disk space usage

Warning: Clearing all data is permanent and cannot be undone. Always verify before executing.

For detailed retention policy information, see the Data Retention Policy Guide.

InfluxDB Status Page

Monitor the health and status of your InfluxDB time-series database.

URL: `https://<ran-monitor-ip>:9443/nokia/influx`

The InfluxDB Status page showing connection status, measurements, batch writer performance, and storage information.

Connection Status

Field	Description
Connection Status	Green indicator when connected, red when disconnected
Database	Configured InfluxDB bucket name
InfluxDB Version	Detected database version (2.x)

Measurements and Data Points

Real-time data point counts for each measurement type:

Measurement	Description
Performance Metrics	PM data points collected from devices
Configuration	Configuration snapshots stored
Alarms	Alarm records in database
Total	Sum of all data points

Batch Writer Performance

Statistics for the InfluxDB batch writer process that handles all data ingestion:

Metric	Description
Queue Size	Points waiting to be written. Color-coded: green (< 1000), yellow (< 10000), orange (< 20000), red (>= 20000)
Filter Rate	Percentage of duplicate data points blocked from writing
PM Filtered	Count of PM counters filtered out (non-dashboard counters not in stored PM data list)
Queue Drops	Points dropped due to queue overflow (should be 0 in normal operation)
Config Cache	Number of unique configuration hashes cached for delta detection
Alarm Cache	Number of active alarms cached for delta detection

Additional metrics:

Metric	Description
Total Written	Cumulative points written to InfluxDB since startup
Flushes	Number of batch flush operations
Filtered	Total duplicate points filtered (not written)
Data Written	Total bytes written to InfluxDB
Throughput	Current write throughput (KB/s or MB/s)
Writer Uptime	Time since batch writer started
Last Flush	Time since last successful flush

Clear Caches: Resets delta detection caches. Use when you want to force re-writing of all data (e.g., after schema changes).

Storage Information

Field	Description
Retention Policies	Current retention settings (default: Indefinite)
Disk Usage	Estimated database size based on record counts
Activity	Last update timestamp

Configuration Details

Field	Description
Host	InfluxDB server hostname
Port	InfluxDB server port (default: 8086)
Bucket	InfluxDB bucket name
Status	Connection status badge
Measurements	Number of measurement types (3: PerformanceMetrics, Configuration, Alarms)

Health Diagnostics

Status indicators for system health:

- **InfluxDB Connectivity** - Database accessible and responding
- **Data Collection** - Performance metrics being collected from devices
- **Data Retention** - Current retention policy status
- **Last Sync** - Most recent data synchronization

Auto-Refresh

The page automatically refreshes every 30 seconds.

Interpreting Status

Condition	Meaning
Connected + Data Counts Growing	System operating normally
Connected + No Data	Check device registration and PM counter configuration
Disconnected	Verify InfluxDB is running and network connectivity
High Queue Size (yellow/red)	InfluxDB write performance issue or network latency
High Queue Drops	Queue overflow - increase batch size or reduce collection rate
High Filter Rate	Good - indicates effective duplicate detection

Use Cases

- Verify InfluxDB is receiving data
- Monitor batch writer health and throughput
- Troubleshoot write performance issues
- Check delta detection cache effectiveness
- Confirm data is being written

System Metrics Page

Real-time performance monitoring for the InfluxDB batch writer and system resources.

URL: `https://<ran-monitor-ip>:9443/nokia/metrics`

The System Metrics page showing InfluxDB batch writer statistics and per-Airscale write counts.

InfluxDB Batch Writer

Summary statistics for the batch writer process:

Metric	Description
Queue Size	Number of data points waiting to be written to InfluxDB
Flush Count	Total number of batch flushes since startup
Total Points Written	Cumulative data points written to InfluxDB
Last Flush	Time since last successful flush operation

InfluxDB Write Statistics by Airscale

Per-device breakdown of data written to InfluxDB:

Column	Description
Airscale	Device name
Performance Metrics	Count of PM data points written
Configuration	Count of configuration snapshots written
Alarms	Count of alarm records written
Total Records	Sum of all data points for this device
Last Write	Timestamp of most recent write for this device

The totals row at the bottom shows aggregate counts across all devices.

System Resources

Erlang VM resource utilization:

Metric	Description
Total Memory	Total memory allocated to the Erlang VM
Process Memory	Memory used by Erlang processes
Binary Memory	Memory used for binary data (XML, JSON payloads)
Atom Memory	Memory used for atoms
Process Count	Number of active Erlang processes
Run Queue	Number of processes waiting for CPU time (0 is healthy)

Auto-Refresh

The page automatically refreshes every 5 seconds.

Use Cases

- Monitor batch writer health and throughput
- Identify devices with high data volume
- Troubleshoot write performance issues
- Verify data is flowing from all devices
- Monitor system resource usage

PM Data Collection Page

Control which Performance Metrics (PM) counters are stored in InfluxDB. Nokia AirScale base stations report over 22,000 unique PM counters, but only a subset are typically needed for dashboards.

URL: `https://<ran-monitor-ip>:9443/nokia/pm-filters`

The PM Data Collection page showing stored counters (left) and available counters (right) with category filters.

How PM Filtering Works

```
eNodeB PM Data (22,000+ counters)
         │
         ▼
┌──────────────────────────────┐
│ PM Counter Whitelist Filter  │
│ Only "Stored PM Data" pass   │
└──────────────────────────────┘
         │
         ▼
    Write Queue
         │
         ▼
      InfluxDB
```

The batch writer filters incoming PM data against the "Stored PM Data" list. Counters not in this list are dropped before queuing, reducing storage and

improving query performance.

Interface Layout

Section	Description
Stored PM Data (Left)	Counters currently being collected and written to InfluxDB
Available Counters (Right)	All 22,000+ counters available to add
Configuration Keys (Bottom)	Config parameters being tracked (from `config_keys.csv`)

Stored PM Data Panel

Counters in this list are written to InfluxDB when received from devices.

Filters:

- **Search**: Filter by counter ID or description
- **Category**: Filter by technology (LTE, 5G-NR, etc.)
- **Source**: Filter by Default (from CSV) or Added (user-added)

Actions:

- **Remove Selected**: Stop collecting selected counters
- **Select All**: Select all visible counters
- **Reset to Defaults**: Restore original list from `pm_counters.csv`

Available Counters Panel

Browse and add counters from the complete Nokia PM reference.

Filters:

- **Search**: Find counters by ID or description keywords
- **Category**: Filter by technology category

Actions:

- **Add Selected**: Start collecting selected counters
- **Select All**: Select all visible counters (limited to 200 displayed)

Counter Categories

Category	Code Prefix	Description
LTE	M8xxx	LTE L1/L2/L3 counters
WCDMA	M5xxx	3G WCDMA counters
5G-NR	M55xxx	5G NR counters
5G-Mobility	M51xxx	5G mobility metrics
5G-Common	M40xxx	5G common counters

Persistence

Changes are:

- **Persisted to disk** in `priv/pm_filters.etf`
- **Survive application restarts**
- **Take effect immediately** (no restart required)

Related Statistics

The InfluxDB Status page shows filtering statistics:

- **PM Filtered**: Count of PM points dropped (not in whitelist)
- **Filter Rate**: Percentage of duplicate config/alarm points blocked

For detailed guidance on counter selection, see PM Data Collection Guide.

Data Management Page

Manage cached data, temporary files, and persistent storage.

URL: `https://<ran-monitor-ip>:9443/nokia/data`

The Data Management page showing ETS cache, temporary files, and InfluxDB data clearing options.

ETS Cache (In-Memory)

Volatile in-memory caches that are cleared on application restart:

Cache	Description
Nokia Config Cache	Cached device configuration data
Nokia Alarms Cache	Cached active alarm records
Retention Record Cache	Cached retention policy record counts
InfluxDB Status Cache	Cached InfluxDB connection status

Clear All ETS Cache: Removes all entries from in-memory caches. Data is regenerated on next request.

Temporary Files

Files created during configuration extraction and processing:

- TAR extracts from device configuration downloads
- Temporary archives created during processing

Clear Temporary Files: Removes temporary files from `/tmp` directory.

InfluxDB Data (Persistent)

Time-series data stored in InfluxDB for each device:

- Performance metrics (PM counters)
- Configuration snapshots
- Alarm records

Per-Device Clearing: Click "Clear Data" next to a device to remove all InfluxDB data for that device.

Clear All: Removes all InfluxDB data for all devices. Use with caution.

Device Registry

Shows count of registered devices. Device configurations are loaded from `config/runtime.exs` at startup and stored in ETS.

Use Cases

- Free up memory by clearing caches
- Clean up temporary files after troubleshooting
- Remove test data before production use
- Clear data for decommissioned devices

Web UI Workflows

Common operational workflows using the Web UI.

Daily Health Check

Purpose: Verify system health at start of shift

Steps:

1. Open main dashboard
2. Verify all devices show green status
3. Check alarm count and severity
4. Review any red/failed devices
5. Investigate issues as needed
6. Document any actions taken

Time: < 5 minutes

Alarm Investigation

Purpose: Respond to and resolve alarms

Steps:

1. Open Alarms page
2. Sort by severity (Critical first)
3. Click alarm for full details
4. Navigate to affected device
5. Cross-reference with recent metrics
6. Determine required action
7. Implement resolution
8. Verify alarm clears

For detailed alarm handling procedures, see the Alarm Management Guide.

Configuration Update

Purpose: Safely update device configuration

Steps:

1. Download current configuration (backup)
2. Modify configuration offline using appropriate tools
3. Upload new configuration to device
4. Note the Plan ID returned
5. Validate configuration using Plan ID
6. If validation succeeds, activate configuration
7. Verify changes took effect
8. Monitor device stability for 15-30 minutes
9. Document change in change management system

Safety Notes:

- Always validate before activating
- Make changes during maintenance windows when possible
- Have rollback plan ready
- Monitor for unexpected behavior

Adding a New Base Station

Purpose: Add newly deployed base station to monitoring

Steps:

1. Check Unconfigured eNodeBs page for device
2. Note Agent ID and IP address
3. Add device to `config/runtime.exs`
4. Restart RAN Monitor application
5. Verify device appears in Base Stations page
6. Confirm registration succeeds (green status)
7. Check metrics are flowing to InfluxDB
8. Set retention policy if needed
9. Add to Grafana dashboards

For detailed operations, see the Common Operations Guide.

Troubleshooting Device Connectivity

Purpose: Diagnose and fix device connection issues

Steps:

1. Check Base Stations page for device status
2. If red/failed, click device for details
3. Review session information and last contact time
4. Check Application Logs for error messages
5. Verify network connectivity (ping device)
6. Confirm credentials are correct
7. Check device is accessible on configured port
8. Review device-side logs if needed
9. Restart connection or device as needed
10. Verify recovery

For detailed troubleshooting, see the Troubleshooting Guide.

Related Documentation

- **Getting Started Guide** - Initial setup and deployment
- **Common Operations Guide** - Day-to-day operational tasks
- **Alarm Management Guide** - Alarm handling and escalation
- **Runtime Configuration Guide** - System configuration
- **Grafana Integration Guide** - Analytics and dashboards
- **Data Retention Policy Guide** - Data lifecycle management
- **Troubleshooting Guide** - Common issues and solutions

RAN Monitor Operations Guide

Radio Access Network (RAN) Monitoring & Management Platform

by Omnitouch Network Services

Table of Contents

Overview

RAN Monitor is a management and monitoring platform for Nokia AirScale base stations in 3GPP LTE and 5G networks. It provides real-time visibility into your RAN equipment health, performance, and configuration.

Key Features

- **Real-Time Monitoring** - Continuous collection of performance metrics and alarms

- **Automated Management** - Maintains persistent connections with base stations
- **Historical Analytics** - Stores data for trend analysis and capacity planning
- **Web Dashboard** - Real-time operational visibility through built-in Web UI
- **Grafana Integration** - Advanced analytics and custom dashboards

System Components

Component	Purpose	Access
RAN Monitor Manager	Core application managing base station connections	Background service
Web UI Control Panel	Real-time operational dashboard	https://<server>:9443
MySQL Database	Session state and device configuration	Internal
InfluxDB	Time-series metrics storage	http://<server>:8086
Grafana	Analytics dashboards and alerting	http://<server>:3000
TCE NSNTI Server	Trace collection from base stations	TCP port 49151
TCE TZSP Forwarder	Real-time trace export to Wireshark	UDP port 37008

Example: Detailed Monitoring Dashboard

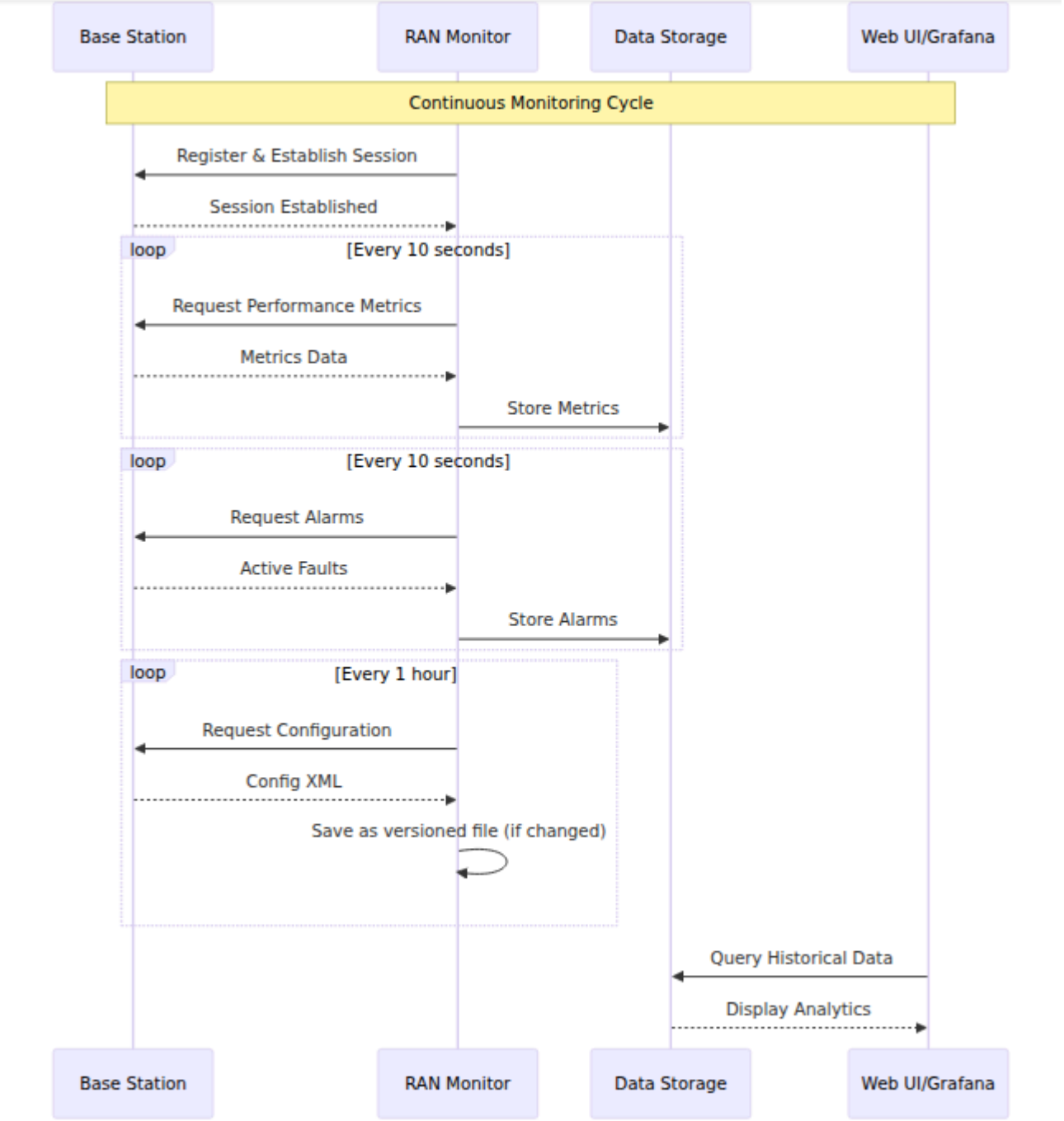
Comprehensive monitoring dashboard showing S1 connections status by LNMME, operational state, data transferred, UEs connected, average PRB usage, performance monitoring metrics, and geographic coverage map. This dashboard provides network operators with at-a-glance visibility into device health, connectivity status, and key performance indicators.

What RAN Monitor Does

RAN Monitor operates continuously in the background to:

1. **Register and Connect** - Establishes secure connections with your Nokia base stations
2. **Collect Performance Data** - Gathers KPIs every 10 seconds (configurable)
3. **Monitor Alarms** - Tracks faults and their severity levels
4. **Track Configuration** - Records system state and parameter changes
5. **Store Historical Data** - Preserves metrics in time-series database
6. **Provide Visibility** - Displays real-time status through Web UI and Grafana

Data Flow



What Gets Collected

Performance Metrics:

- Cell availability and uptime
- Traffic throughput (uplink/downlink)
- Resource utilization (PRB usage)

- Call setup success rates
- Handover performance
- Radio quality measurements

Alarms:

- Fault severity (Critical, Major, Minor, Warning)
- Affected systems and components
- Probable cause and descriptions
- Timestamps and durations

Configuration:

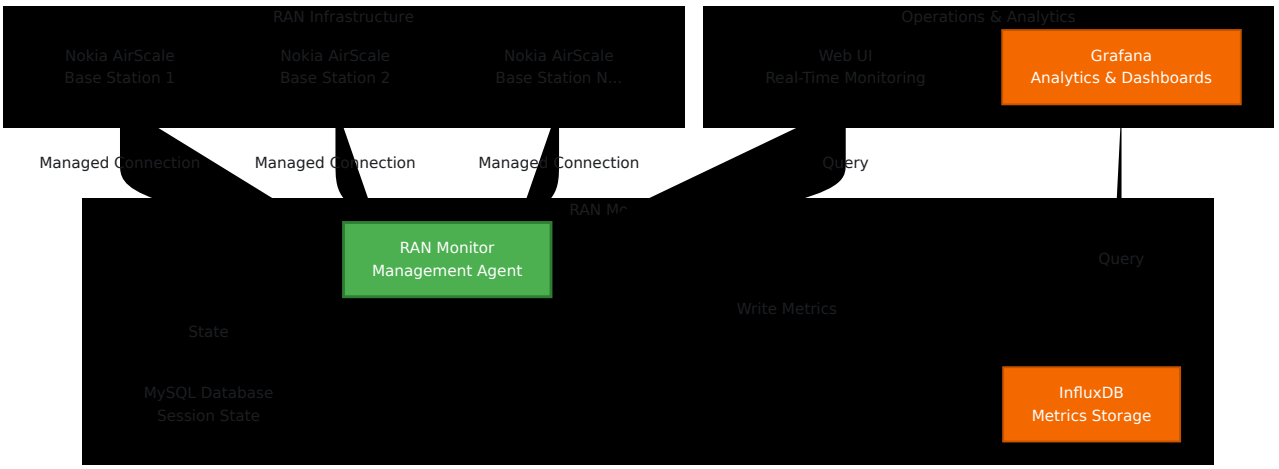
- Complete XML configuration snapshots (stored as versioned files)
- Automatic change detection and versioning
- Configuration history and audit trail
- Last 10 versions retained per device

For config management details, see the Configuration Archive Guide.

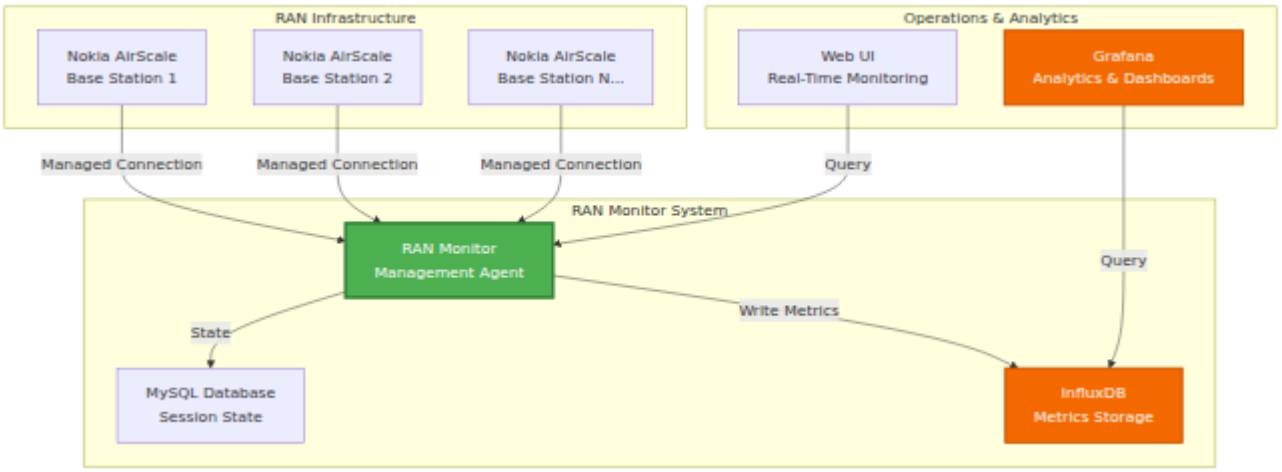
For detailed counter definitions, see the Nokia Counter Reference.

System Architecture

Infrastructure Overview



Configuration Overview



For complete configuration details, see the Runtime Configuration Guide.

Trace Collection Entity (TCE)

RAN Monitor includes an integrated Trace Collection Entity for capturing and analyzing LTE/5G protocol messages. This enables detailed troubleshooting, drive testing, and RF optimization.

What is TCE?

The Trace Collection Entity receives trace data from Nokia AirScale base stations containing:

- **S1-AP Messages** - Control plane signaling between eNodeB and EPC
- **RRC Messages** - Radio Resource Control signaling
- **NAS Messages** - Non-Access Stratum signaling
- **User Plane Data** - PDCP layer throughput information

Use Cases

Drive Testing:

- Capture end-user RF experience
- Analyze handover performance

- Measure signal quality (RSRP, RSRQ, SINR)
- Identify coverage holes

Troubleshooting:

- Debug call setup failures
- Analyze handover issues
- Investigate dropped calls
- Review mobility events

RF Optimization:

- PCI planning validation
- Neighbor relation optimization
- Handover parameter tuning
- Coverage and capacity analysis

For complete trace collection procedures and Wireshark analysis, see the **TCE MDT Data Collection Guide**.

Web UI Overview

RAN Monitor includes a built-in Web UI for real-time operational monitoring and management.

Access: `https://<ran-monitor-ip>:9443`

The main dashboard provides at-a-glance visibility into system health, device status, and active alarms.

Key Pages

Main Dashboard

Real-time system overview with:

- System health indicators

- Device status summary (associated/failed counts)
- Active alarm counts by severity
- Recent activity and events

Auto-refreshes every 5 seconds for real-time visibility.

Base Stations Page

View all managed devices with their current status:

- Connection status (green = associated, red = failed)
- Registration state and session information
- Last contact timestamp
- Filter, search, and sort capabilities

Click any device to view detailed information including registration details, session lifecycle, recent metrics, and active alarms.

Alarms Page

Monitor all faults across your network:

- Color-coded by severity (Red = Critical, Orange = Major, Yellow = Minor, Blue = Warning, Green = Cleared)
- Alarm details, probable cause, affected system
- Timestamps and duration tracking
- Sort by severity and filter capabilities

For alarm handling procedures, see the Alarm Management Guide.

Configuration Management

Safely manage base station configurations:

1. **Download** current configuration (backup)
2. **Upload** new configuration → receive Plan ID
3. **Validate** configuration using Plan ID
4. **Activate** validated configuration
5. **Verify** changes took effect

Always validate before activating to prevent service interruptions.

Configuration Archive: All configuration changes are automatically tracked and versioned. View historical configs, download previous versions, or compare changes via the Config Archive page.

For detailed procedures, see the Web UI Guide - Configuration Management and Configuration Archive Guide.

Unconfigured eNodeBs

Discover base stations attempting to connect that aren't yet configured:

- Agent ID (use when adding to configuration)
- Last seen timestamp
- Number of connection attempts
- Actions: Refresh, Delete, Clear All

Use Case: When new base stations are deployed, they appear here. Copy the Agent ID and add them to `config/runtime.exs`.

Application Logs

Real-time logging dashboard for troubleshooting:

- Filter by log level (Emergency through Debug)
- Search across all messages
- Pause/Resume live streaming
- Dynamically change system log level
- Color-coded by severity

For troubleshooting procedures, see the Troubleshooting Guide.

Data Retention Policy

Manage how long data is stored in InfluxDB:

- View global retention policy and total record counts
- Set per-device retention periods
- View record counts by measurement type (Performance Metrics, Configuration, Alarms)
- Manually trigger cleanup or clear all data for a device

For complete data retention information, see the Data Retention Policy Guide.

InfluxDB Status

Monitor time-series database health:

- Connection status indicator
- Measurement counts by type
- Storage information
- Database version and configuration
- Auto-refreshes every 5 minutes

Interpreting Status:

- Connected + Growing Counts = Normal operation
- Connected + No Data = Check device registration
- Disconnected = Check InfluxDB connectivity

Complete Web UI Guide

For comprehensive Web UI documentation including all features, workflows, and best practices, see:

Web UI Guide - Complete control panel reference

Monitoring with Grafana

While the Web UI provides real-time visibility, Grafana enables deep historical analysis and custom dashboards.

Why Use Grafana?

Grafana is Best For:

- Historical trend analysis over days/weeks/months

- Custom KPI dashboards tailored to your needs
- Long-term capacity planning
- Pattern identification and anomaly detection
- Executive reporting and SLA tracking
- Advanced alerting with notification channels

Web UI is Best For:

- Immediate device status checks
- Real-time alarm monitoring
- Configuration management
- Session troubleshooting
- System administration tasks

Example Grafana dashboard showing cell availability, throughput trends, and resource utilization

Dashboard Types

Executive Summary Dashboard:

- Network-wide health overview
- Total alarm count by severity

- Average cell availability across all sites
- Aggregate throughput and capacity metrics
- Device status grid

NOC Operations Dashboard:

- Real-time active issues table
- Resource utilization gauges
- Traffic overview (last 24 hours)
- Alarm trend charts
- Device status quick view

Engineering Deep-Dive Dashboard:

- Traffic pattern analysis
- Cell quality metrics (SINR, RSRP distributions)
- Radio performance (RLC retransmission, RRC setup success)
- Configuration audit trail
- Correlation analysis

Nokia AirScale Performance Dashboard:

- PRB utilization (DL/UL)
- Throughput trends (PDCP layer)
- Active UE counts
- Cell availability calculations
- Per-cell resource breakdowns
- RSSI measurements
- RRC connection setup success
- VSWR by antenna
- Power consumption

For complete dashboard examples, query patterns, and counter definitions, see:

Grafana Integration Guide - Complete analytics and dashboarding guide

---

Common Operations

Daily Operations

Daily Health Check (5-10 minutes):

1. Open Web UI dashboard
2. Verify all devices show green status
3. Check alarm count and severity
4. Review any failed devices
5. Investigate issues as needed

For detailed procedures, see the Web UI Guide - Workflows.

Alarm Investigation:

1. Open Alarms page, sort by severity
2. Click alarm for full details
3. Navigate to affected device
4. Cross-reference with metrics
5. Determine required action and resolve

For alarm handling procedures, see the Alarm Management Guide.

Device Management

Adding a New Base Station:

1. Verify network connectivity to device
2. Check Unconfigured eNodeBs page for device
3. Add device to `config/runtime.exs`
4. Restart RAN Monitor

5. Verify registration succeeds (green status)
6. Confirm metrics flowing to InfluxDB

Removing a Base Station:

1. Decide whether to preserve or delete historical data
2. Comment out or remove device from `config/runtime.exs`
3. Optionally clear data via Data Retention page
4. Restart RAN Monitor
5. Update Grafana dashboards

Updating Device Credentials:

1. Note current device status
2. Update credentials in `config/runtime.exs`
3. Restart RAN Monitor
4. Verify reconnection succeeds

For complete operational procedures, see:

Common Operations Guide - Day-to-day management tasks

Configuration Management

Safe Configuration Update Workflow:

1. **Download** current configuration (backup) - or retrieve from Config Archive
2. **Modify** configuration offline
3. **Upload** to device → get Plan ID
4. **Validate** using Plan ID → verify no errors
5. **Activate** if validation succeeds
6. **Verify** changes took effect
7. **Monitor** device stability for 15-30 minutes
8. **Confirm** new version appears in Config Archive (within 1 hour)

Important: Always validate before activating. Schedule changes during maintenance windows when possible.

Configuration Rollback: If issues occur, download a previous version from Config Archive and upload it using the same workflow.

For base station configuration details, see the AirScale Configuration Guide.

For configuration history and versioning, see the Configuration Archive Guide.

Documentation Index

RAN Monitor documentation is organized by audience and use case:

For Operations Teams (NOC, Administrators)

Document	Purpose	When to Use
Web UI Guide	Complete control panel reference	Daily operations, monitoring devices
Common Operations Guide	Day-to-day management tasks	Adding devices, managing configs, backups
Configuration Archive Guide	Config versioning and history	Tracking config changes, rollback, auditing
Alarm Management Guide	Alarm handling and escalation	Investigating faults, responding to alerts
Troubleshooting Guide	Problem resolution procedures	When issues occur, error diagnosis
Data Retention Policy Guide	Data lifecycle management	Managing storage, setting retention periods

For Engineering and Analytics

Document	Purpose	When to Use
Grafana Integration Guide	Dashboards, queries, and alerting	Building dashboards, setting up alerts
Nokia Counter Reference	Performance counter definitions	Understanding metrics, creating queries
AirScale Configuration Guide	Base station setup and configuration	Configuring devices, understanding parameters
TCE MDT Data Collection Guide	MDT trace collection and Wireshark analysis	Collecting drive test data, coverage optimization
API Endpoints Reference	REST API documentation	Integrations, automation, scripting

For Configuration and Deployment

Document	Purpose	When to Use
Runtime Configuration Guide	Complete configuration reference	Initial setup, modifying settings

Quick Start

New to RAN Monitor?

1. Start with Web UI Guide to learn the interface
2. Review Common Operations Guide for routine tasks
3. Study Alarm Management Guide for alarm handling

4. Keep Troubleshooting Guide bookmarked for issues

Setting Up Monitoring?

1. See Grafana Integration Guide for dashboards
2. Reference Nokia Counter Reference for metrics
3. Review Data Retention Policy Guide for storage management

Quick Reference

Access Points

Service	URL	Purpose
Web UI Dashboard	`https://<server>:9443`	Real-time monitoring and management
Grafana	`http://<server>:3000`	Analytics dashboards and alerts
InfluxDB	`http://<server>:8086`	Metrics database (usually internal access only)

Important Paths

Path	Purpose
`config/runtime.exs`	Main configuration file (devices, databases, settings)
`priv/cert/`	SSL certificates for HTTPS Web UI
`priv/external/nokia/`	Manager authentication keys
`priv/airscale_configs/`	Configuration archive (versioned XML files)

Key Concepts

Session Management:

- RAN Monitor establishes sessions with base stations
- Sessions have expiry times and require keep-alive
- Re-registration happens automatically (default: every 30 seconds)
- Session state stored in MySQL database

Data Flow:

- Metrics collected every 10 seconds (configurable)
- Alarms collected every 10 seconds via polling + real-time webhooks
- Configuration snapshots every 1 hour (saved as versioned files when changed)
- Performance metrics and alarms written to InfluxDB for historical storage

Data Retention:

- Global default: 720 hours (30 days)
- Per-device overrides available
- Automatic cleanup runs hourly
- Manual cleanup available via Web UI

For configuration details, see the Runtime Configuration Guide.

Common Workflows

Daily Health Check:

1. Open Web UI → Dashboard
2. Check device status (all green?)
3. Review alarm count
4. Investigate any issues

Respond to Critical Alarm:

1. Web UI → Alarms → Sort by severity
2. Click alarm for details
3. Navigate to device
4. Review recent metrics and config changes
5. Implement resolution
6. Verify alarm clears

Add New Device:

1. Verify network connectivity
2. Edit `config/runtime.exs`
3. Add device to airscales list
4. Restart RAN Monitor
5. Verify registration (green status)

Support

Troubleshooting Resources

Resource	Use For
Troubleshooting Guide	Common issues and solutions
Application Logs Page	Real-time system logs and errors
Device Detail View	Session status, registration issues
InfluxDB Status Page	Data collection verification

Quick Diagnostic Steps

Device Not Connecting:

1. Check Base Stations page → device status
2. Verify network connectivity: `ping <device-ip>`
3. Check credentials in `config/runtime.exs`
4. Review Application Logs for errors

No Metrics in Grafana:

1. Check device is associated (green status)
2. Verify InfluxDB Status page shows growing counts
3. Test InfluxDB connectivity
4. Check Grafana data source configuration

Web UI Not Loading:

1. Verify port 9443 is accessible
2. Check firewall allows HTTPS traffic
3. Verify SSL certificates exist
4. Review application logs for Web UI startup errors

For complete troubleshooting procedures, see the Troubleshooting Guide.

Getting Help

Before Contacting Support:

Gather this information:

- Problem description and when it started
- Error messages from Application Logs
- Affected devices (names/IPs)
- Recent configuration changes
- RAN Monitor version and OS

Contact:

For assistance with RAN Monitor:

- Omnitouch Network Services support
- Include gathered diagnostic information
- Provide configuration files (sanitize passwords)
- Include relevant log excerpts

Self-Service:

1. Search the Troubleshooting Guide
2. Check Application Logs for specific errors
3. Review recent configuration changes
4. Test connectivity and basic functionality
5. Consult relevant documentation guides

Documentation Map