

# Infrastructure health monitoring — specification

Infrastructure health monitoring — SPEC R0 (normative)

| Field                  | Value                                               |
|------------------------|-----------------------------------------------------|
| Author                 | Anton Afanasyeu                                     |
| Revision               | R0                                                  |
| Creation date          | 2026-06-26                                          |
| Last modification date | 2026-06-26                                          |
| Co-authored            | Cursor Agent (project assistant)                    |
| Severity               | high                                                |
| State                  | accepted                                            |
| Document type          | spec                                                |
| Supersedes             | —                                                   |
| Design review          | docs/DRs/20260626_cluster_monitoring.md (frozen R0) |

## Table of contents

1. Purpose
2. Scope
3. Requirements
  - 3.1 Must
  - 3.2 Should
  - 3.3 Non-goals
4. Monitoring domains
  - 4.1 Hardware metrics
  - 4.2 Software and service metrics
  - 4.3 External availability
  - 4.4 Provider-layer fault escalation
5. Topology — cast04 as dedicated monitoring node
6. Component catalog
  - 6.1 Alpine apk packages (cast04)
  - 6.2 Alpine apk packages (cast01–cast03)
  - 6.3 Docker-based (cast04)
  - 6.4 External SaaS (no install)
7. Deployment plan
  - 7.1 Phase 0 — cast04 provisioning
  - 7.2 Phase 1 — Prometheus + node\_exporter
  - 7.3 Phase 2 — Service probes and MariaDB exporter
  - 7.4 Phase 3 — Alertmanager
  - 7.5 Phase 4 — Grafana dashboards
  - 7.6 Phase 5 — Uptime Kuma + UptimeRobot
  - 7.7 ac-deploy integration
8. Alert routing
  - 8.1 Alert severity levels
  - 8.2 Routing rules
  - 8.3 Alertmanager SMTP config (skeleton)
  - 8.4 Telegram receiver config (skeleton)
  - 8.5 Alert rule catalog
9. Grafana dashboard catalog
10. Pre-requisite accounts
11. TimescaleDB relationship
12. Verification and acceptance
13. Risks and mitigations
14. Related documents
15. Changelog

Source: 20260626\_cluster\_monitoring.md — section links work in PDF viewers that support internal anchors.

**Document type:** SPEC

**Source draft:** docs/drafts/20260626\_cluster\_monitoring.txt

**Design review:** docs/DRs/20260626\_cluster\_monitoring.md (frozen)

**PDF:** 20260626\_cluster\_monitoring.pdf · Regenerate: `bash ac-docs/build-pdf.sh`

**Status:** **Accepted** — implement per §7 deployment plan

**Scope:** Passive infrastructure monitoring and alerting for XEN cluster cast01–cast04; Track A only

**Related:** INFRA.md · DRs/20260626\_cluster\_monitoring.md · specs/20100612\_1\_scaling.md ·

20260620\_postgres\_timescale\_db\_platform.md

**Track B (future):** drafts/20260626\_cluster\_orchestration.txt

## 1. Purpose

Define the **normative** implementation of passive infrastructure monitoring and alerting for the AndroidCast XEN-based VM cluster. The system must:

- Record hardware and software health metrics from all cluster nodes continuously
- Alert operators promptly when measured values exceed defined thresholds
- Provide visual dashboards for trend analysis and incident investigation
- Detect external (provider-layer) unavailability via an independent probe

This SPEC covers **Track A only** (observe, record, alert). Automated cluster remediation is Track B and is explicitly out of scope — see drafts/20260626\_cluster\_orchestration.txt.

## 2. Scope

| In scope                                               | Out of scope                                   |
|--------------------------------------------------------|------------------------------------------------|
| Hardware metrics: CPU, RAM, disk, NIC per node         | Automated node provisioning / deprovisioning   |
| Software metrics: process health, HTTP probes, MariaDB | Log aggregation (Grafana Loki — planned R1)    |
| Alert routing: email + Telegram                        | Auto-scaling or auto-failover reactions        |
| External uptime probe (UptimeRobot)                    | Changes to FE nginx or router software         |
| Grafana dashboards (infra layer)                       | Business / application metrics (→ TimescaleDB) |
| cast04 as dedicated monitoring VM                      | Kubernetes / service mesh                      |
| ac-deploy integration (monitoring role)                | SaaS external aggregator for metric data       |
| Pre-requisite account setup guide                      | iOS / mobile performance profiling             |

**Alpha constraint:** Monitoring stack MAY be deployed incrementally per §7 phases. Each phase MUST leave prior phases operational.

## 3. Requirements

### 3.1 Must

| ID       | Requirement                                                                              |
|----------|------------------------------------------------------------------------------------------|
| R-MON-1  | cast04 dedicated monitoring VM provisioned as Alpine LTS paravirt XEN guest              |
| R-MON-2  | node_exporter running on cast01, cast02, cast03, cast04                                  |
| R-MON-3  | Prometheus scraping all node_exporters on 15s interval                                   |
| R-MON-4  | blackbox_exporter probing all critical HTTP endpoints (§4.2) from cast04                 |
| R-MON-5  | mysqld_exporter running on MariaDB primary node                                          |
| R-MON-6  | Alertmanager routing CRITICAL alerts to email within 2 min of threshold breach           |
| R-MON-7  | Grafana serving dashboards on cast04 (internal access via SSH tunnel at R0)              |
| R-MON-8  | UptimeRobot external HTTP probe active for apps.f0xx.org                                 |
| R-MON-9  | All software installed via Alpine apk or Docker (no source compilation)                  |
| R-MON-10 | No new software installed on FE VM or router                                             |
| R-MON-11 | Metric data retained locally on cast04 for minimum 90 days                               |
| R-MON-12 | Alert for CPU steal time > 10% sustained 5 min (Xen overcommit indicator → notify Anton) |
| R-MON-13 | Alert for disk write latency > 100ms sustained 5 min on MariaDB node                     |
| R-MON-14 | Alert for MariaDB replication lag > 30s                                                  |
| R-MON-15 | Hardware guarantee baseline alerts (§4.1 table) — notify when measured < guaranteed      |

### 3.2 Should

| ID       | Requirement                                                                                                              |
|----------|--------------------------------------------------------------------------------------------------------------------------|
| R-MON-S1 | Alertmanager Telegram receiver configured alongside email                                                                |
| R-MON-S2 | Uptime Kuma status page deployed on cast04 (Docker)                                                                      |
| R-MON-S3 | Grafana provisioned with community dashboard for node_exporter (ID 1860)                                                 |
| R-MON-S4 | Grafana provisioned with community dashboard for MariaDB                                                                 |
| R-MON-S5 | Prometheus metric retention configurable via <code>--storage.tsdb.retention.time</code>                                  |
| R-MON-S6 | Alertmanager grouping configured to suppress alert storms ( <code>group_wait</code> 30s, <code>group_interval</code> 5m) |
| R-MON-S7 | ac-deploy monitoring/ role added to cluster0 scripts for reproducible deployment                                         |

### 3.3 Non-goals

- Automated node scale-out or scale-in (Track B)
- DDoS automatic mitigation (Track B)
- Storage automatic failover (Track B)
- Centralized log search (Grafana Loki — R1)
- Business metrics dashboards (TimescaleDB + Grafana — separate track)
- Monitoring of router or FE internal metrics (config-only access; UptimeRobot covers external availability)

## 4. Monitoring domains

### 4.1 Hardware metrics

Collected by **node\_exporter** on every VM (cast01–cast04). Alert thresholds are guidelines — tune after collecting baseline.

| Metric              | Prometheus metric name                                                                                 | WARNING threshold | CRITICAL threshold | Xen note                      |
|---------------------|--------------------------------------------------------------------------------------------------------|-------------------|--------------------|-------------------------------|
| CPU utilization     | <code>rate(node_cpu_seconds_total{mode!="idle"}[5m])</code>                                            | > 80% (5 min)     | > 95% (5 min)      |                               |
| CPU steal time      | <code>rate(node_cpu_seconds_total{mode="steal"}[5m])</code>                                            | > 5% (5 min)      | > 10% (5 min)      | Xen overcommit → notify Anton |
| Load average 1m     | <code>node_load1 / node_cpu_count</code>                                                               | > 1.5             | > 3.0              |                               |
| RAM used %          | <code>1 - node_memory_MemAvailable_bytes / node_memory_MemTotal_bytes</code>                           | > 80%             | > 90%              |                               |
| Swap used %         | <code>node_memory_SwapFree_bytes / node_memory_SwapTotal_bytes</code>                                  | > 50%             | > 80%              |                               |
| Disk used %         | <code>1 - node_filesystem_avail_bytes / node_filesystem_size_bytes</code>                              | > 75%             | > 90%              |                               |
| Disk write latency  | <code>rate(node_disk_write_time_seconds_total[5m]) / rate(node_disk_writes_completed_total[5m])</code> | > 50ms            | > 100ms            | Critical on MariaDB node      |
| Disk IOPS           | <code>rate(node_disk_io_time_seconds_total[5m])</code>                                                 | > 80%             | > 95%              |                               |
| NIC ingress bytes/s | <code>rate(node_network_receive_bytes_total[5m])</code>                                                | > 150 Mbit/s      | > 190 Mbit/s       | Guaranteed: 200 Mbit/s        |
| NIC egress bytes/s  | <code>rate(node_network_transmit_bytes_total[5m])</code>                                               | > 150 Mbit/s      | > 190 Mbit/s       | Guaranteed: 200 Mbit/s        |
| NIC packet drops    | <code>rate(node_network_receive_drop_total[5m])</code>                                                 | > 0.1%            | > 1%               |                               |
| NIC errors          | <code>rate(node_network_receive_errs_total[5m])</code>                                                 | > 0               | > 10/min           |                               |

**Hardware guarantee baseline check:** If `node_memory_MemTotal_bytes` < 7.5 GB, or CPU core count < 8, or disk total < 30 GB → CRITICAL alert (guaranteed resources not provided → notify Anton).

### 4.2 Software and service metrics

#### Process / systemd health (node\_exporter systemd collector or blackbox TCP probe)

| Service                        | Check                 | Alert trigger                                       |
|--------------------------------|-----------------------|-----------------------------------------------------|
| nginx                          | TCP :80 probe         | Connection refused → CRITICAL                       |
| php-fpm                        | TCP socket probe      | No response → CRITICAL                              |
| MariaDB                        | TCP :3306 probe       | Connection refused → CRITICAL                       |
| sshd                           | TCP :22 probe         | Connection refused → WARNING                        |
| Per-microservice HTTP endpoint | HTTP GET /health or / | Non-2xx or timeout > 5s → WARNING; > 30s → CRITICAL |

#### MariaDB metrics (mysqld\_exporter)

| Metric                 | Prometheus metric                                                                                   | WARNING      | CRITICAL     |
|------------------------|-----------------------------------------------------------------------------------------------------|--------------|--------------|
| Replication lag        | mysql_slave_status_seconds_behind_master                                                            | > 10s        | > 30s        |
| Open connections       | mysql_global_status_threads_connected                                                               | > 80% of max | > 95% of max |
| Slow queries/s         | rate(mysql_global_status_slow_queries[5m])                                                          | > 1/s        | > 5/s        |
| InnoDB buffer pool hit | mysql_global_status_innodb_buffer_pool_reads / mysql_global_status_innodb_buffer_pool_read_requests | < 95%        | < 90%        |
| Aborted connections    | rate(mysql_global_status_aborted_connects[5m])                                                      | > 1/min      | > 10/min     |

#### nginx metrics (via node\_exporter nginx stub or blackbox HTTP probe)

| Metric            | Source                            | WARNING          | CRITICAL          |
|-------------------|-----------------------------------|------------------|-------------------|
| 5xx response rate | blackbox HTTP probe returning 5xx | Any 5xx returned | > 10% of requests |
| Response time     | blackbox probe_duration_seconds   | > 2s             | > 10s             |

#### php-fpm metrics (via php-fpm status endpoint + blackbox or custom exporter)

| Metric                  | WARNING | CRITICAL |
|-------------------------|---------|----------|
| Active workers % of max | > 70%   | > 90%    |
| Queue length            | > 5     | > 20     |

## 4.3 External availability

**Tool:** UptimeRobot free tier — no installation; checks from UptimeRobot's global nodes.

| Monitor      | URL                                            | Check interval | Alert on           |
|--------------|------------------------------------------------|----------------|--------------------|
| Frontend TLS | https://apps.f0xx.org                          | 5 min          | Non-2xx or timeout |
| API health   | https://apps.f0xx.org/app/androidcast_project/ | 5 min          | Non-2xx or timeout |

**Internal cross-check:** blackbox\_exporter on cast04 also probes the FE endpoint from inside the cluster. If UptimeRobot fires but blackbox does NOT → provider/uplink fault (Anton's layer). If both fire → likely cluster-side issue.

## 4.4 Provider-layer fault escalation

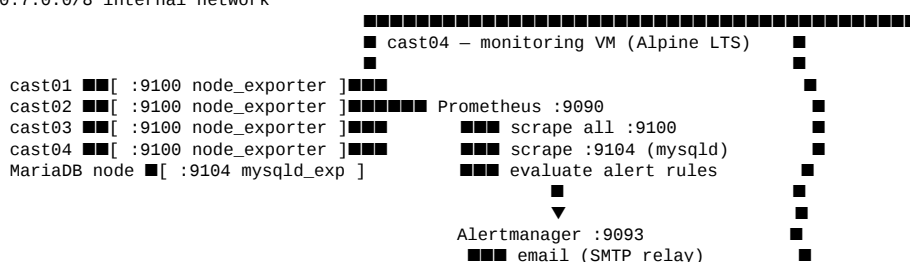
When monitoring detects that the root cause is in Anton's infrastructure layer:

| Symptom                                       | Likely cause              | Action                           |
|-----------------------------------------------|---------------------------|----------------------------------|
| CPU steal > 10% sustained                     | Xen host overcommit       | Email Anton; await response      |
| Disk write latency persistent despite low I/O | Xen host disk contention  | Email Anton                      |
| NIC saturation below 200 Mbit guarantee       | Uplink / Xen host NIC     | Email Anton                      |
| UptimeRobot alert + cast04 unreachable        | Xen host or uplink down   | Email Anton; check intra-network |
| Entire cluster unreachable                    | Router / uplink / payment | Email Anton; check calendar      |

These events are **alert-only** at R0. No automated remediation.

## 5. Topology — cast04 as dedicated monitoring node

10.7.0.0/8 internal network





## 7.1 Phase 0 — cast04 provisioning

**Goal:** cast04 VM exists and is reachable from cast01–cast03 on internal network.

| Action       | Detail                                                     |
|--------------|------------------------------------------------------------|
| Request VM   | Ask Anton for cast04 — spec per §5 table                   |
| Base install | Alpine latest LTS; configure apk repos (community + edge)  |
| SSH access   | Add developer SSH key; test login                          |
| Network      | Confirm cast04 can reach cast01–03 IPs on internal network |
| NTP          | apk add chrony && rc-service chrony start                  |

**Verify:** ssh cast04 ping cast01 succeeds; apk update succeeds.

## 7.2 Phase 1 — Prometheus + node\_exporter

**Goal:** Prometheus scraping hardware metrics from all nodes.

apk add prometheus prometheus-node-exporter

```
global:
  scrape_interval: 15s
  evaluation_interval: 15s
```

```
scrape_configs:
- job_name: 'node'
  static_configs:
    - targets:
      - 'cast01:9100'
      - 'cast02:9100'
      - 'cast03:9100'
      - 'cast04:9100'
```

```
rc-update add prometheus default
rc-update add node-exporter default
rc-service prometheus start
rc-service node-exporter start
```

On cast01–cast03: install and start prometheus-node-exporter (§6.2).

**Verify:** curl http://cast04:9090/api/v1/targets shows all 4 nodes in state up.

## 7.3 Phase 2 — Service probes and MariaDB exporter

**Goal:** HTTP/TCP probes and MariaDB metrics flowing.

Add to prometheus.yml:

```
- job_name: 'blackbox_http'
  metrics_path: /probe
  params:
    module: [http_2xx]
  static_configs:
    - targets:
      - https://apps.f0xx.org
      - http://cast01:80 # nginx on workload nodes
      # add per-microservice /health endpoints as they exist
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: cast04:9115 # blackbox_exporter

- job_name: 'blackbox_tcp'
  metrics_path: /probe
  params:
    module: [tcp_connect]
  static_configs:
    - targets:
      - cast01:3306 # MariaDB
      - cast02:3306 # replica
      - cast01:22
      - cast02:22
      - cast03:22
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: cast04:9115

- job_name: 'mariadb'
  static_configs:
    - targets: ['<mariadb-primary-node>:9104']
```

Start services:

```
rc-service blackbox-exporter start
rc-service mysqld-exporter start # on MariaDB node
```

**Verify:** curl "http://cast04:9115/probe?target=https://apps.f0xx.org&module=http\_2xx" returns probe\_success 1.

## 7.4 Phase 3 — Alertmanager

**Goal:** Alerts route to email and (optionally) Telegram on threshold breach.

See §8 for full Alertmanager config skeletons. Deploy skeleton, then fill in SMTP credentials and Telegram token.

```
apk add alertmanager
rc-update add alertmanager default
rc-service alertmanager start
```

Add alerting: and rule\_files: blocks to prometheus.yml (see §8.5 for rule file examples).

**Verify:** amtool --alertmanager.url=http://cast04:9093 alert returns empty (no active alerts); send a test alert and confirm email receipt.

## 7.5 Phase 4 — Grafana dashboards

**Goal:** Visual dashboards accessible for all metrics collected in phases 1–3.

```
apk add grafana
rc-update add grafana default
rc-service grafana start
```

In Grafana UI:

1. Add Prometheus datasource: http://localhost:9090
  2. Import community dashboard **1860** ("Node Exporter Full") — covers §4.1 metrics
  3. Import community dashboard **7362** or **13106** (MariaDB / mysqld\_exporter)
  4. Create custom panel for blackbox probe results (status per service endpoint)
  5. Create overview panel showing hardware guarantee baseline vs measured values
- Verify:** All four dashboards render data; node\_exporter CPU steal panel shows non-zero (or zero — both valid).

## 7.6 Phase 5 — Uptime Kuma + UptimeRobot

**Goal:** Service status page + external probe active.

```
rc-service docker start
docker run -d --restart=always --name uptime-kuma \
  -p 3001:3001 -v uptime-kuma:/app/data louislam/uptime-kuma:latest
```

Configure monitors in Uptime Kuma UI:

- Add HTTP monitors for each microservice /health endpoint
- Add TCP monitors for MariaDB :3306, nginx :80
- Configure notification to same email as Alertmanager

**UptimeRobot:**

- Register free account at https://uptimerobot.com
- Add HTTP(s) monitor: https://apps.f0xx.org, 5-minute interval
- Add email alert notification (same ops email as Alertmanager)

**Verify:** Uptime Kuma dashboard shows all green; UptimeRobot sends test notification successfully.

## 7.7 ac-deploy integration

Add a monitoring/ role to cluster0 deploy scripts in ac-deploy/sim/cluster0/:

```
ac-deploy/sim/cluster0/
monitoring/
  deploy-monitoring.sh      # installs cast04 components
  prometheus.yml.j2         # Jinja2/envsubst template with node IPs
  alertmanager.yml.j2       # template with SMTP/Telegram creds from env
  alert-rules/
    hardware.yml            # §4.1 rules
    services.yml            # §4.2 rules
  grafana/
    datasources.yml         # auto-provision Prometheus datasource
    dashboards/             # provisioned dashboard JSONs
```

Credentials (SMTP password, Telegram token) MUST be passed via environment variables or a secrets file outside git — never committed.

**Verify:** bash deploy-monitoring.sh on a fresh cast04 produces a running stack within 10 minutes.

## 8. Alert routing

### 8.1 Alert severity levels

| Level    | Meaning                                 | Response                                          |
|----------|-----------------------------------------|---------------------------------------------------|
| critical | Service down or hardware limit exceeded | Alert immediately; wake operator if outside hours |
| warning  | Approaching threshold; degraded         | Alert during business hours; batch in digest      |
| info     | Notable event; no action required       | Log only; no alert                                |

### 8.2 Routing rules

```
critical → email (immediately) + Telegram (immediately)
warning  → email (grouped, 5 min window)
info     → no alert (recorded in Prometheus only)
```

### 8.3 Alertmanager SMTP config (skeleton)

```
global:
  smtp_smarthost: 'smtp.gmail.com:587'      # or Mailgun: smtp.mailgun.org:587
```

```

smtp_from: 'alerts@your-domain.com'
smtp_auth_username: 'your-gmail@gmail.com'
smtp_auth_password: '<APP_PASSWORD>' # from env / secrets file
smtp_require_tls: true

route:
  group_by: ['alertname', 'instance']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 4h
  receiver: 'ops-email'
  routes:
    - match:
        severity: critical
      receiver: 'ops-critical'
      group_wait: 10s
      repeat_interval: 1h

receivers:
- name: 'ops-email'
  email_configs:
    - to: 'ops@your-domain.com'
      send_resolved: true

- name: 'ops-critical'
  email_configs:
    - to: 'ops@your-domain.com'
      send_resolved: true
  telegram_configs:
    - bot_token: '<TELEGRAM_BOT_TOKEN>' # from env / secrets file
      chat_id: '<TELEGRAM_CHAT_ID>'
      send_resolved: true

inhibit_rules:
- source_match:
    severity: 'critical'
  target_match:
    severity: 'warning'
  equal: ['instance']

```

**Note:** Replace <APP\_PASSWORD> and <TELEGRAM\_BOT\_TOKEN> with values from environment — never commit credentials to git.

## 8.4 Telegram receiver config (skeleton)

Obtain Telegram credentials:

1. Open Telegram, message @BotFather
2. /newbot → follow prompts → save TOKEN
3. Add the bot to your ops group or DM it, then: `curl https://api.telegram.org/bot<TOKEN>/getUpdates` → find chat.id
4. Set TELEGRAM\_BOT\_TOKEN=<TOKEN> and TELEGRAM\_CHAT\_ID=<CHAT\_ID> in the secrets file used by deploy script

## 8.5 Alert rule catalog

Create /etc/prometheus/rules/hardware.yml:

```

groups:
- name: hardware
  rules:
    - alert: CpuHighUtilization
      expr: 100 - (avg by(instance) (rate(node_cpu_seconds_total{mode="idle"}[5m])) * 100) > 95
      for: 5m
      labels:
        severity: critical
      annotations:
        summary: "CPU critical on {{ $labels.instance }}"
        description: "CPU utilization {{ $value | humanize }}% > 95%"

    - alert: XenCpuStealHigh
      expr: avg by(instance) (rate(node_cpu_seconds_total{mode="steal"}[5m])) * 100 > 10
      for: 5m
      labels:
        severity: critical
      annotations:
        summary: "Xen CPU steal critical on {{ $labels.instance }} – notify Anton"
        description: "CPU steal {{ $value | humanize }}% > 10% – hypervisor overcommit"

    - alert: RamCritical
      expr: (1 - node_memory_MemAvailable_bytes / node_memory_MemTotal_bytes) * 100 > 90
      for: 5m
      labels:
        severity: critical
      annotations:
        summary: "RAM critical on {{ $labels.instance }}"

    - alert: DiskCritical
      expr: (1 - node_filesystem_avail_bytes{fstype!~"tmpfs|overlay"} / node_filesystem_size_bytes) * 100 > 90
      for: 5m
      labels:
        severity: critical
      annotations:
        summary: "Disk critical on {{ $labels.instance }} mount {{ $labels.mountpoint }}"

    - alert: DiskWriteLatencyHigh
      expr: >

```

```

    rate(node_disk_write_time_seconds_total[5m])
    / rate(node_disk_writes_completed_total[5m]) * 1000 > 100
for: 5m
labels:
  severity: critical
annotations:
  summary: "Disk write latency critical on {{ $labels.instance }} – possible HW fault; notify Anton"

- alert: HardwareGuaranteeRamBreach
  expr: node_memory_MemTotal_bytes < 7.5 * 1024 * 1024 * 1024
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "RAM below guaranteed 8 GB on {{ $labels.instance }} – notify Anton"

```

Create /etc/prometheus/rules/services.yml:

```

groups:
- name: services
  rules:
    - alert: ServiceDown
      expr: probe_success == 0
      for: 1m
      labels:
        severity: critical
      annotations:
        summary: "Service unreachable: {{ $labels.instance }}"

    - alert: ServiceSlowResponse
      expr: probe_duration_seconds > 10
      for: 2m
      labels:
        severity: warning
      annotations:
        summary: "Slow response from {{ $labels.instance }}: {{ $value | humanize }}s"

    - alert: MariaDbReplicationLag
      expr: mysql_slave_status_seconds_behind_master > 30
      for: 2m
      labels:
        severity: critical
      annotations:
        summary: "MariaDB replication lag {{ $value | humanize }}s on {{ $labels.instance }}"

    - alert: MariaDbDown
      expr: mysql_up == 0
      for: 1m
      labels:
        severity: critical
      annotations:
        summary: "MariaDB exporter cannot reach database on {{ $labels.instance }}"

```

Add to prometheus.yml:

```

rule_files:
- "/etc/prometheus/rules/*.yaml"

```

alerting:

```

  alertmanagers:
  - static_configs:
    - targets: ['cast04:9093']

```

## 9. Grafana dashboard catalog

| Dashboard                       | Grafana ID | Source    | Covers                                        |
|---------------------------------|------------|-----------|-----------------------------------------------|
| Node Exporter Full              | 1860       | community | CPU, RAM, disk, NIC, load — all \$4.1 metrics |
| MariaDB Overview                | 7362       | community | Connections, queries, replication lag, InnoDB |
| Blackbox Exporter               | 7587       | community | HTTP probe success/latency per target         |
| Custom: Hardware Guarantee      | —          | create    | Measured vs Anton's guaranteed values         |
| Custom: Service Health Overview | —          | create    | All service probes in one panel grid          |

**Provisioning via ac-deploy:** export dashboard JSON from Grafana UI after customization and store in ac-deploy/sim/cluster0/monitoring/grafana/dashboards/. Add datasource YAML to grafana/datasources/prometheus.yml.

## 10. Pre-requisite accounts

The following must be created by the project owner before Phase 3 deploy:

| Account            | Service                                   | Purpose                 | Notes                                                                       |
|--------------------|-------------------------------------------|-------------------------|-----------------------------------------------------------------------------|
| Gmail app password | Gmail Settings → Security → App passwords | Alertmanager SMTP relay | App password, not account password. Alternatively: Mailgun free (10k/month) |

|                  |                         |                                |                                          |
|------------------|-------------------------|--------------------------------|------------------------------------------|
| Telegram bot     | @BotFather in Telegram  | Alertmanager Telegram receiver | /newbot → TOKEN;<br>getUpdates → CHAT_ID |
| UptimeRobot free | https://uptimerobot.com | External HTTP probe            | Add monitor for<br>https://apps.f0xx.org |

**Important:** ImprovMX and addy.io provide inbound email routing (MX records) — they are not outbound SMTP relay services and cannot be used with Alertmanager. Use Gmail app password or a dedicated transactional email service (Mailgun, SendGrid) for outbound alerts.

## 11. TimescaleDB relationship

This SPEC implements infra-layer monitoring. Business and application metrics are a separate concern handled by the TimescaleDB platform plan (20260620\_postgres\_timescale\_db\_platform.md).

| Layer                  | Store                     | Examples                                   | Retention                |
|------------------------|---------------------------|--------------------------------------------|--------------------------|
| Infrastructure         | Prometheus TSDB on cast04 | CPU steal, disk latency, service probe     | 90 days                  |
| Business / application | TimescaleDB (PostgreSQL)  | Crash rate, session count, graph analytics | Long-term (configurable) |

Grafana on cast04 can query both stores simultaneously. No ETL or data duplication is required between them. The TimescaleDB datasource is added to Grafana as a second datasource in a future step — it does not affect R0 monitoring deployment.

## 12. Verification and acceptance

All phases must pass their gate before the next phase begins.

| Phase     | Acceptance criteria                                                                                             |
|-----------|-----------------------------------------------------------------------------------------------------------------|
| 0         | cast04 reachable from cast01–03; apk update succeeds                                                            |
| 1         | curl http://cast04:9090/api/v1/targets shows all 4 nodes state=up                                               |
| 2         | Blackbox probe for https://apps.f0xx.org returns probe_success 1; mysqld_exporter metrics visible in Prometheus |
| 3         | Test alert fires and email is received within 2 minutes; Telegram notification received (if configured)         |
| 4         | Grafana dashboard 1860 renders data for all nodes; MariaDB dashboard shows replication status                   |
| 5         | Uptime Kuma shows all services green; UptimeRobot sends test notification                                       |
| ac-deploy | Fresh cast04 reproduced by running deploy-monitoring.sh within 10 minutes                                       |

**Ongoing regression:** After any cluster change (new microservice, new VM, nginx config change), add corresponding blackbox probe target and Grafana panel within 48 hours.

## 13. Risks and mitigations

| Risk                                   | Mitigation                                                                                                                                               |
|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| cast04 itself fails — monitoring blind | UptimeRobot external probe continues; Alertmanager writes state to disk and resumes after VM restart                                                     |
| XEN host failure takes cast04 offline  | UptimeRobot fires from external — operator is alerted even without cast04                                                                                |
| Alpine edge package breaks Prometheus  | Pin version in /etc/apk/world; test upgrades on cast04 before rolling to workload nodes                                                                  |
| SMTP rate limit during incident storm  | Alertmanager group_interval: 5m + repeat_interval: 4h limit volume; Telegram is parallel channel                                                         |
| Prometheus disk overflow               | 32 GB cast04 disk; estimated < 5 GB for 3 nodes at 15s scrape over 90 days; monitor cast04 disk in Grafana                                               |
| Credentials leaked in git              | Secrets must be in environment variables or a secrets file excluded from git (.gitignore); never in prometheus.yml or alertmanager.yml committed to repo |
| FE/router metrics unavailable          | FE nginx stub_status available via config if Anton allows; not mandatory — blackbox HTTP probe sufficient at R0                                          |

## 14. Related documents

| Document                                  | Role                                       |
|-------------------------------------------|--------------------------------------------|
| DRs/20260626_cluster_monitoring.md        | Frozen design review (source of this SPEC) |
| drafts/20260626_cluster_orchestration.txt | Track B — auto-remediation placeholder     |
| INFRA.md                                  | FE/BE topology, hosts, ports               |

|                                            |                                                         |
|--------------------------------------------|---------------------------------------------------------|
| specs/20100612_1_scaling.md                | VM roles and cluster layout                             |
| specs/20260618_repos_reorganizing.md       | Microservice split (each new MS needs a blackbox probe) |
| 20260620_postgres_timescale_db_platform.md | Business metrics store (complements Prometheus)         |
| GRAFANA_vs_others_graphvis_pivot.md        | Earlier Grafana evaluation for application analytics    |
| orchestration/sim/cluster0/ARCHITECTURE.md | Lab cluster node layout                                 |

## 15. Changelog

| Rev | Date       | Change                                                                                      |
|-----|------------|---------------------------------------------------------------------------------------------|
| R0  | 2026-06-26 | SPEC converted from draft + DR R0; all decisions locked; deployment plan §7; alert rules §8 |