

Infrastructure health monitoring — design review (DR)

Infrastructure health monitoring — frozen DR R0

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	DR
Superseded by	—
Specification	docs/specs/20260626_cluster_monitoring.md

Table of contents

1. Executive summary
2. Problem statement
3. Track A / Track B scope split
4. Responsibility domains
5. Monitoring topology — cast04
6. Tool selection
 - 6.1 Decided stack
 - 6.2 Options rejected
 - 6.3 Future additions (R1 / Track B)
7. Alert channels
8. Infrastructure constraints
9. Relationship to TimescaleDB / business metrics
10. Risks and mitigations
11. Non-goals (Track B reference)
12. Related documents
13. Changelog

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

Document type: DR (Design Review)

Source draft: docs/drafts/20260626_cluster_monitoring.txt

Specification: docs/specs/20260626_cluster_monitoring.md (normative implementation doc)

PDF: 20260626_cluster_monitoring.pdf · Regenerate: bash ac-docs/build-pdf.sh

Status: Accepted R0 — implement per SPEC

Scope: Passive infrastructure monitoring and alerting for the XEN-based VM cluster (cast01–cast04); Track A only — automated remediation is Track B (drafts/20260626_cluster_orchestration.txt)

Related: INFRA.md · specs/20100612_1_scaling.md · specs/20260618_repos_reorganizing.md · orchestration/sim/cluster0/ARCHITECTURE.md · 20260620_postgres_timescale_db_platform.md

1. Executive summary

Question: How should the VM cluster (cast01–cast03) and its microservices be continuously monitored, and what tooling should be deployed?

Short answer: Deploy a dedicated monitoring VM (cast04, Alpine LTS) running Prometheus + node_exporter (per-node) + blackbox_exporter + mysqld_exporter + Alertmanager + Grafana. Alerts route to email (SMTP relay) and Telegram. External availability is covered by UptimeRobot (free tier, no install). Automated remediation is out of scope for R0 — it becomes Track B.

DR decisions (R0):

Item	Decision
Scope	Track A: passive monitoring + alerting only; no auto-remediation
Monitoring node	cast04 — dedicated 4th Alpine LTS VM; paravirt XEN guest
Metrics stack	Prometheus + node_exporter + blackbox_exporter + mysqld_exporter
Dashboards	Grafana (community dashboards; self-hosted on cast04)
Alerting	Alertmanager → email SMTP relay + Telegram bot API
External uptime	UptimeRobot free tier — no install; HTTP probe to apps.f0xx.org from internet
Status page	Uptime Kuma on cast04 (Docker) — internal, access via SSH tunnel
Log aggregation	Out of scope R0; Grafana Loki planned for R1 (Track B)
VictoriaMetrics	Not used at R0 — not in Alpine apk; standard Prometheus sufficient at 3-node scale
Zabbix	Not chosen — batteries-included but inflexible for future business metrics integration
Nagios	Not chosen — legacy, superseded by Prometheus+Alertmanager
Kibana / ELK	Excluded — log-centric, heavy, wrong tool for infra metrics
FE / router changes	None at R0 — monitoring is internal; FE nginx config change only if external Grafana access added later
DNS changes	None at R0
Data residency	All metric data stays self-hosted (no external aggregator)
Track B	Placeholder draft created: drafts/20260626_cluster_orchestration.txt

2. Problem statement

The project runs a cluster of XEN-based VMs (cast01–cast03) hosting microservices at various complexity levels — from a single service per node to multi-node DB master/replica chains. As the microservice split progresses (per specs/20260618_repos_reorganizing.md), manual observation of each service and node becomes impractical.

Key failure modes that must be detected and alerted:

Category	Example failure
Hardware	CPU steal time spiking (Xen overcommit), disk SMART pre-failure, RAM exhaustion
Software	nginx 5xx surge, php-fpm worker saturation, MariaDB replication lag, process crash
Network	Egress/ingress saturation, NIC packet drops
Provider	Uplink outage, XEN host failure, VM unavailability

None of these are currently monitored in a structured way.

3. Track A / Track B scope split

This DR and its SPEC cover **Track A only**.

Track	Scope	Milestone
Track A (this SPEC)	Passive monitoring: observe, record, alert	R0
Track B (future)	Active remediation: orchestration reacts to alert signals — auto-scale, auto-failover, DDoS response	R1+

Track B details and prerequisites are captured in drafts/20260626_cluster_orchestration.txt. Track B MUST NOT be designed until Track A alerting is stable and at least 4 weeks of baseline metric data has been collected.

4. Responsibility domains

The infrastructure has two clearly bounded ownership layers:

■ Anton's layer (SaaS – contact owner for issues)	■
■ • Internet provider / uplink	■
■ • Router (Debian; config-only from project side)	■
■ • XEN hypervisor hosts + VM provisioning	■
■ • Internal network (10.7.0.0/8), git server	■
■ • FE VM (Gentoo HVM; nginx TLS; config-only)	■
=====	
delegated cluster boundary ↓	
■ Project owner's layer (full control)	■
■ • cast01 / cast02 / cast03 – workload VMs	■
■ • cast04 – monitoring VM (this SPEC)	■
■ • All software installed on these VMs	■
■ • MariaDB, nginx, PHP-FPM, microservices	■
=====	

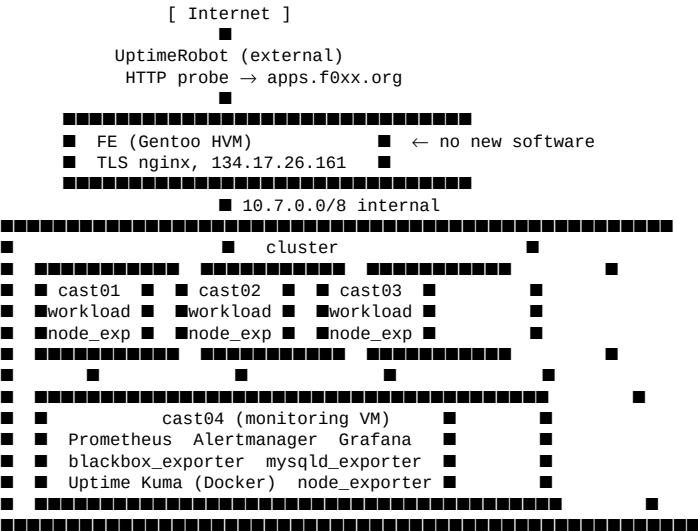
Monitoring responsibility boundary: The project owner monitors everything within the delegated cluster. When monitoring detects that the cause is in Anton's layer (hardware fault, uplink, XEN host instability), the action is to notify Anton — not to auto-remediate.

Hardware guarantee baseline (from Anton's SLA commitment per draft):

Resource	Guaranteed value
CPU	Intel Xeon Gold 5218, 8 real cores @ 2.30 GHz per node
RAM	8 GB per node
Disk	32 GB non-shared per node
Network	200 Mbit/s in/out
Cluster size	3 workload nodes (cast01–cast03)

Monitoring MUST alert when measured values diverge significantly from this baseline (e.g. CPU steal > 10% for > 5 min = Xen overcommit alarm → notify Anton).

5. Monitoring topology — cast04



- Prometheus scrapes node_exporter on cast01/cast02/cast03/cast04 via internal IPs
- blackbox_exporter on cast04 probes service HTTP/TCP endpoints from inside cluster
- mysql_exporter runs on the MariaDB primary node (cast01 or cast02 per deployment)
- cast04 has no workload services — monitoring-only
- Access to Grafana/Uptime Kuma UI: SSH tunnel from dev machine at R0

cast04 minimum spec (request from Anton):

Resource	Minimum
CPU	2 vCPU
RAM	2 GB (Prometheus + Grafana + Alertmanager)
Disk	32 GB (90 days metric retention at 15s scrape)
OS	Alpine latest LTS, paravirt XEN

6. Tool selection

6.1 Decided stack

Component	Package / source	Role
Prometheus	prometheus (apk)	Time-series scraper + TSDB
node_exporter	prometheus-node-exporter (apk)	Hardware + OS metrics per VM
blackbox_exporter	prometheus-blackbox-exporter (apk)	HTTP/TCP service health probes
mysqld_exporter	prometheus-mysqld-exporter (apk)	MariaDB metrics
Alertmanager	alertmanager (apk)	Alert routing (email + Telegram)
Grafana	grafana (apk)	Dashboards
Uptime Kuma	Docker image louislam/uptime-kuma	Service status page
UptimeRobot	SaaS free tier (web account)	External HTTP probe

All apk packages are available in Alpine community/edge repos. Docker (docker apk) is available for Uptime Kuma.

6.2 Options rejected

Tool	Reason not chosen
Kibana / ELK	Log-centric, heavy (Elasticsearch needs 4+ GB), wrong tool for infra metrics
Zabbix	Batteries-included but less composable; harder to extend to business metrics later
Nagios Core	Legacy; status-only; no time-series; superseded by Prometheus+Alertmanager
VictoriaMetrics	Not in Alpine apk repos; standard Prometheus sufficient at 3-node scale
Grafana Cloud	Data must remain self-hosted (PO preference)
Netdata	Good for quick bootstrap but adds another agent daemon; Prometheus covers the same ground

6.3 Future additions (R1 / Track B)

Component	Role
Grafana Loki + Promtail	Centralized log aggregation (replaces manual syslog review)
Alertmanager webhook handlers	Trigger ac-deploy scripts on alert (foundation for Track B remediation)

7. Alert channels

Channel	Tool	Requirement
Email	Alertmanager SMTP	Required — primary alert channel
Telegram	Alertmanager Telegram receiver	Recommended — fast ops notifications
Status page	Uptime Kuma	Recommended — visual per-service uptime
External uptime	UptimeRobot	Required — provider-fault detection

Email SMTP relay: Alertmanager needs an outbound SMTP relay (NOT an inbound routing service). Options: Gmail account + app password, or Mailgun free tier (10k emails/month). **Note:** ImprovMX and addy.io are inbound email routing services — they are not SMTP relay providers and cannot be used for Alertmanager outbound alerts.

Telegram: Purely API-based — no software installation. Register a bot via @BotFather, obtain TOKEN and CHAT_ID (add bot to ops group or DM). Alertmanager calls Telegram HTTP API from cast04.

Accounts to create before deploy (PO action):

Account	Purpose	Notes
Gmail app password OR Mailgun free	Alertmanager SMTP relay	Gmail: Settings → Security → App passwords
Telegram bot via @BotFather	Alertmanager Telegram receiver	Note TOKEN + CHAT_ID
UptimeRobot free account	External HTTP probe	uptimerobot.com; add monitor for apps.f0xx.org

8. Infrastructure constraints

Constraints discovered during DR review — all accounted for in SPEC:

Constraint	Impact
Paravirt XEN guest (no HVM)	node_exporter works fine on paravirt; Xen steal time metric available
Alpine LTS + apk only	All stack components available as apk packages; Uptime Kuma via Docker (apk-installable)
FE / router: no new software	Monitoring is entirely internal; no FE changes at R0
Data self-hosted	No Grafana Cloud or external aggregator
8 GB RAM per workload node	Standard Prometheus on cast04 is sufficient (Prometheus ~200–500 MB for this scale)

9. Relationship to TimescaleDB / business metrics

The project has a separate PostgreSQL + TimescaleDB platform plan (20260620_postgres_timescale_db_platform.md). These two stores serve different purposes and coexist:

Layer	Store	Metrics type	Retention
Infrastructure	Prometheus TSDB (cast04)	CPU, RAM, disk, latency, process health	90 days
Business / app	TimescaleDB (PostgreSQL)	Crash rates, session counts, graph analytics	Long-term

Grafana on cast04 can query **both** data sources. No ETL between them is required.

10. Risks and mitigations

Risk	Mitigation
cast04 itself fails — monitoring goes dark	cast04 is monitored by UptimeRobot externally; Alertmanager persistence via local disk
Xen host failure takes cast04 offline	UptimeRobot external probe survives; operator is alerted from outside
SMTP relay rate limit hit during incident storm	Alertmanager grouping + repeat_interval configured; Telegram as parallel channel
apk package version lag	Use Alpine edge repo for monitoring components if LTS versions are too old
Prometheus disk fill (90 day retention + 3 nodes)	Estimated < 5 GB for 3 nodes at 15s scrape; 32 GB disk leaves comfortable margin
cast04 RAM insufficient	2 GB minimum; Prometheus + Grafana + Alertmanager combined ~500 MB–1 GB typical
External UptimeRobot free tier limits	Free tier: 50 monitors, 5-min check interval — sufficient for R0

11. Non-goals (Track B reference)

The following are explicitly **out of scope** for this DR and SPEC:

- Automatic horizontal scaling (add node under CPU load)
- Automatic scale-in (remove idle node)
- DDoS automated response
- Automatic storage failover
- Any orchestration that modifies cluster topology without human approval

These scenarios are documented as future work in **drafts/20260626_cluster_orchestration.txt** (Track B). Track B DR and SPEC will be authored once Track A is operational and baseline metrics are available.

12. Related documents

Document	Role
specs/20260626_cluster_monitoring.md	Normative implementation SPEC (this DR's output)
drafts/20260626_cluster_orchestration.txt	Track B placeholder (auto-remediation, future)
INFRA.md	FE/BE topology, hosts, ports
specs/20100612_1_scaling.md	VM roles and cluster layout
specs/20260618_repos_reorganizing.md	Microservice split (monitoring must track each MS)
20260620_postgres_timescale_db_platform.md	TimescaleDB for business metrics (complements Prometheus)
orchestration/sim/cluster0/ARCHITECTURE.md	Lab cluster node layout
GRAFANA_vs_others_graphvis_pivot.md	Earlier Grafana evaluation for app analytics

13. Changelog

Rev	Date	Change
R0	2026-06-26	Initial DR — accepted; decisions locked; SPEC generated