Product Roadmap for Observability: Alerting, Logging, and Monitoring

Author: *Shreyas Gune (sgune@protonmail.com)*

Vision:
To deliver an observability platform that enhances the reliability of the system, mitigates downtime, and reduces time to detect issues – which would then aid in faster resolution. We will achieve this through intelligent and comprehensive logging, monitoring and alerting.

Stakeholders Involved and Function

Role	Department	Function
Observability PM	Product	Oversee the overall roadmap, ensure alignment with business goals.
Engineering Load/Architect	Engineering	Lead technical evaluation and tool selection.
DevOps	Ops/SRE	Implement log aggregation, metrics collection, and basic alerting.
SRE	Ops/SRE	Ensure system reliability during setup and integration of monitoring.
Security Lead	Security Engineering	Ensure observability systems are secure, manage access controls.
Customer Support/Success	Customer Success	Provide feedback from customers on system performance issues.
Data Engineer/Analyst	Data Engineering	Work on advanced metrics collection and dashboards
UX/UI Designer	Product/Design	Refine dashboard design based on feedback.
AI/ML Specialist	Machine Learning and AI	Develop and optimize AI/ML models for predictive analytics.

Phase 1: Discovery and Setup (Month 1 and 2)

Objective: Establish foundational observability features to enable basic monitoring, logging, and alerting.

Feature	**Description**	**Estimated Timeline**
Tool Selection and Setup	● Evaluate tool candidates: Prometheus Stack, ELK stack ● Paid: ○ New Relic ○ Datadog	Month 1
Centralized Log aggregation	Implement log aggregation centralize logs across services (Fluentd or Loki or ELK stack)	Month 1-2
Basic Metric Collection (Alerting Metrics as well as Debugging Metrics)	● Instrument key metrics(CPU,Memory, Disk, Networking, DBs) for health monitoring. ● Services need to expose the said metrics on a known standard port, at path “/metrics”	Month 1-2
Threshold-Based Alerting	● Identify relevant metrics to alert on ● Set up Alerting rules in config as code ● Establish Thresholds based on ○ Warning ○ Critical	Month 1-2
Initial Dashboards	● Survey existing dashboards available in the open source community ● Establish queries that render trends on relevant business metrics and KPIs ● Instrument panels in MVP dashboards.	Month 3

Notes to iterate on:
‘

Phase 2: Standardization and Optimization (Month 3 and 4)

Objective: Optimize the observability tools and processes, ensuring consistency, reliability, and actionable insights.

Feature	**Description**	**Estimated Timeline**
Log Format Standardization	Standardize log formats for all services to keep things consistent ● JSON ● Key-value Pairs	Month 3
SLOs and SLIs	● Identify key metrics to create SLOs out of that are relevant to the business goals ● Establish SLOs ● Instrument SLIs in services to said SLOs	Month 3
Correlations	RCA analysis by correlating logs and metrics	Month 4
Alert Tuning	Based on incidents, we refine alerts ● Check for false-positives ● Retune thresholds	Month 4
Incident Management	Integrate incident management tools ● PagerDuty ● Opsgenie ● VictorOps ● xMatters ● AlertManager hooks ● Slack/Teams integration ● JIRA integration	Month 4

Notes to iterate on:

Phase 3: Advanced Insights and Automation (Month 5 and 6)

Objective: Introduce advanced analytics and automation to streamline monitoring and incident resolution.

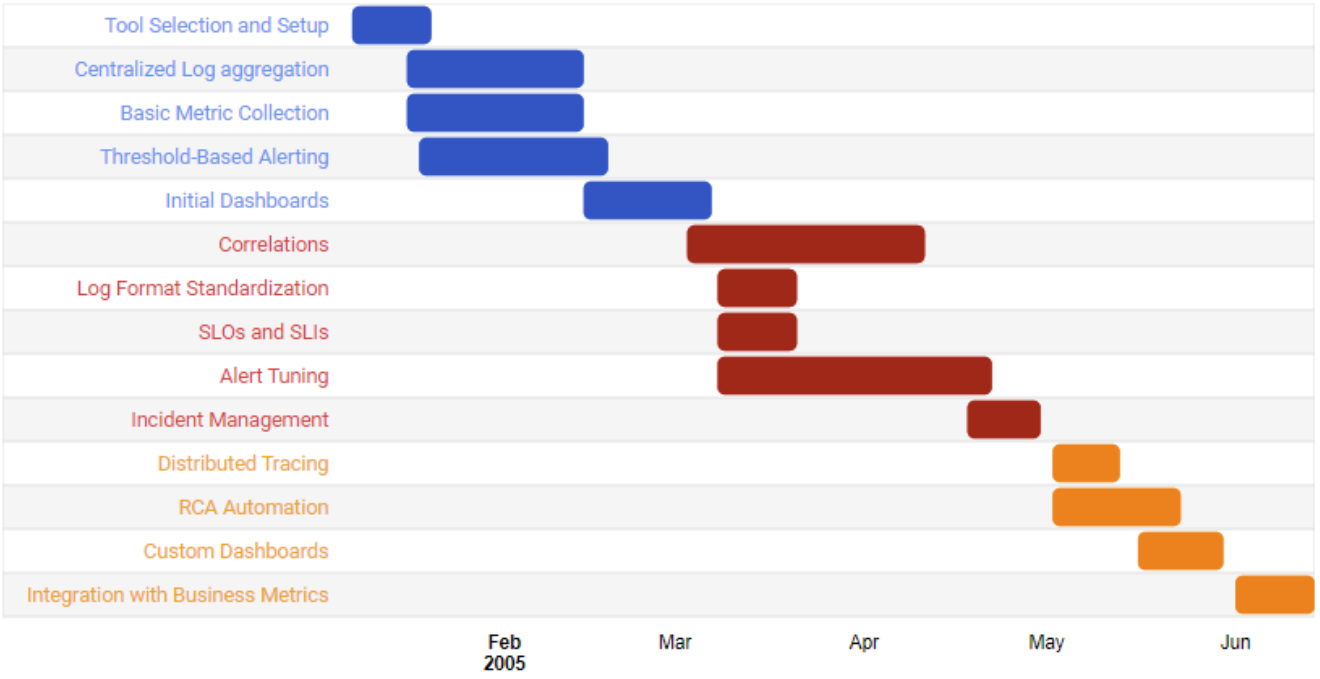
Feature	**Description**	**Estimated Timeline**
Distributed Tracing	Look into implementing tracing using ● OpenTelemetry ● Jaeger or ● Tempo	Month 5
RCA Automation	● Incidents should get pooled into post-mortem doc, which would then link an RCA hub ● Relevant links to ○ Dashboard panels in time-window ○ Logs ○ Traces ○ Ad-hoc Queries	Month 5
Custom Dashboards	Dashboard Tailored to different teams ● DevOps ● Product ● Sales ● Executive ● Customer Facing (controlled via IAM and LDAP)	Month 6
Integration with Business Metrics	Link observability tools with business KPIs (e.g., user engagement, transaction success rates) for broader insights.	Month 6

Notes to iterate on:

Roadmap Summary

Phase	Key Milestones	Duration (month #)
1: Discovery and Setup	• Log aggregation, • Basic metrics, • Threshold-based alerting, • Initial dashboards.	1-3
2. Standardization and Optimization	• Log format standardization, • SLIs & SLOs, • Log-metrics correlation, • Incident management integration, • Alert sensitivity tuning.	3-4
3. Advanced Analytics and Automation	• Distributed tracing, • Root Cause Analysis Automation, • Advanced Dashboards, • Business Intelligence	5-6

Author: Shreyas Gune (sgune@protonmail.com)



Gantt Chart and Table, available at: [Gantt Chart and Table link]

Future work:

Objective: List of nice to haves and goals to work towards to improve and iterate on existing observability stack.

Feature	Description	Additional Notes
Anomaly Detection	Introduce anomaly detection to identify abnormal patterns in metrics/logs. ● Using machine learning models ● Models need to be compliant with security standards and trained in-house	
Predictive Analytics	Develop predictive models to detect potential incidents before they happen ● Service Degradation ● Catastrophic Outages ● Cascading Failure States	
Automated Incident Response	Implement auto-remediation actions for certain alerts. ● Scale services automatically based on load ● Traffic Cutovers ● Version rollbacks and roll forwards	
Intelligent Alerting	Introduce AI-driven intelligent alerting to ● Reduce false positives ● Prioritize critical incidents.	
Self-Healing Systems	Implement self-healing mechanisms ● Auto-scaling ● Service restarts upon anomaly detection	
Enhanced Predictive Analytics	Extend predictive models to predict system failures and performance degradation in real-time.	