Databricks Job Alerts & Secure Log Access

Proactive Monitoring and Role-Based Access Control with Tags

Why This Matters

- Ensure quick detection of job issues
- • Avoid resource waste from long-running jobs
- • Prevent unauthorized access to team-specific logs
- • Promote ownership and isolation

Job Alerts – Failures and Runtime

- Monitor job failures by job ID or tag
- • Track runtime duration to detect stuck jobs
- • Log shipping to Datadog for central monitoring
- • Datadog alerts configured with team tags

Example – Job Failure Alert

- Condition: Job status = 'FAILED'
- • Group by: team or job name tag
- • Trigger: >2 failures in 15 mins
- • Notify: Alert mapped to team email/Slack

Example – Long Running Job Alert

- Condition: Runtime > 60 minutes
- • Track: Start and end timestamps
- • Alert: Missing 'end' event after threshold
- • Use case: Identify stuck or inefficient jobs

Challenge – Log Access Isolation

- Logs shipped to Datadog index
- • All teams by default can access all logs
- • Need isolation by team

Solution – Tag-Based Access Control

- Tag clusters with `team:<name>`
- • Logs routed to Datadog with these tags
- • Restrict Datadog index access using RBAC filters
- • Teams only see logs matching their tags

Implementation Example

- Cluster tag: team:ml
- • Logs tagged and stored in Datadog with team:ml
- • Datadog RBAC policy: Only ML team role can query team:ml logs
- • Other teams restricted from accessing logs outside their scope

Benefits

- • Reduced cross-team noise in logs
- • Data protection between business units
- • Improved observability and compliance

Next Steps

- Finalize tag schema (team, env, app)
- • Enforce tagging on job clusters
- • Apply Datadog RBAC filters on index
- • Deploy alert templates for standard use

Q&A

- Open discussion
- • Feedback on alert thresholds and access model