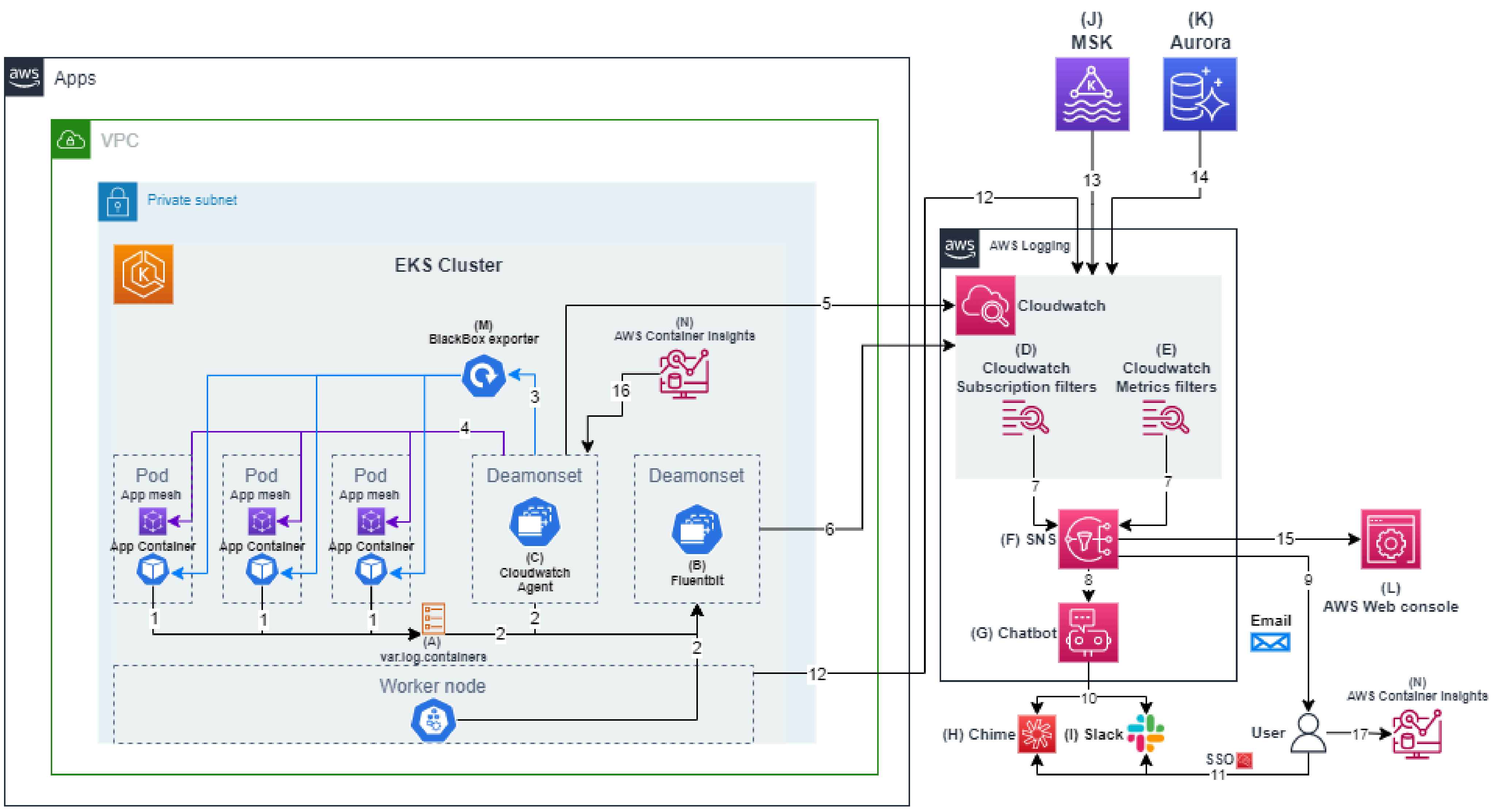
Architecture_for_monitoring_and_observability

Last edited by **czerniga-gft** 8 months ago

Architecture for monitoring and observability

Diagram of Architecture for Monitoring and observalitity



Relevant elements of architecture

- A – var.log.containers - file that contains logs from the app containers pods
- B – Fluentbit (deamon set) is processing the logs (i.e. concatenate, bitrate filter, parse json)
- C – Cloudwatch agent (deamon set) is the middleman between EKS Cluster and Cloudwatch
- D – Cloudwatch Subscription filters scan logs for a specific **phrases, tokens** and forward the matches to a destination of choice (in this case SNS) for alerting, processing or storage. Each log group can have two subscription filters (i.e. it will alert once for each found phrase)
- E – Cloudwatch Metrics filters scan logs for a specific **patterns** and forward the matches to a destination of choice (in this case SNS) for alerting, processing or storage. (i.e. it will alert once for specific number of reapeated logs)
- F – SNS - Simple notification service
- G – AWS Chatbot - creates alerts based on SNS notifications
- H – Chime - Communications service for teams
- I – Slack - Communications service for teams
- J – MSK - Fully managed service for Kafka
- K – Aurora - Fully managed relational database
- L – AWS Web console
- M – BlackBox exporter
- N – AWS Container Insights

Description of architecture for logging

- 1 – Collecting Logs from pods locally in var.log.containers **(A)** file
- 2 – Forwarding the application logs, worker nodes metrics and cloudwatch agent logs to fluentbit (deamon set) **(B)**

  - 6 – Sending the logs and metrics further from Fluentbit (B) to Cloudwatch outside the EKS Cluster. There, filters **(D)** and **(E)** are applied to Cloudwatch log groups.
  - 7 – Filtered logs metrics are passed to SNS **(F) **
  - 8 – SNS **(F)** is passing the notifications to AWS Chatbot **(G) **
  - 9 – SNS **(F)** is passing the notifications to users by email
  - 10 – AWS Chatbot **(G)** is forwarding alerts to either AWS Chime **(H)** or Slack **(I)**
  - 11 – Human users can use SSO to get access to AWS Chime **(H)** or Slack **(I)**

- ## Description of architecture for monitoring

  - 12 – EKS Nodes are triggering Cloudwatch alerts by sending its metrics
  - 13 – MSK **(J)** is triggering Cloudwatch alerts by sending its metrics
  - 14 – Aurora **(K)** is triggering Cloudwatch alerts by sending its metrics
  - 15 – SNS **(F)** forwards alerts to AWS Web Console **(L)**

- ## Description of architecture for application performance monitoring

  - 3 – Cloudwatch Agent **(C)** is querying App coinainers if at least one of them is up and running (it is done through BlackBox exporter **(M)**) (alarm - Service not responding)
  - 4 – Cloudwatch Agent **(C)** is querying App meshes about latency statistics (alarm - too many https errors)
  - 5 – Cloudwatch Agent **(C)** is sending performance logs to Cloudwatch in AWS Logging (alarm - CPU utilization too high)
  - 16 – AWS Container Insights **(N)** is exporting metrics) to Cloudwatch agent **(C) **(alarm - Pod restarts too many
  - 17 – AWS Container Insights **(N)** is available for users in admin's panel

Attachments:

Monitoring Diagram.drawio.png (image/png) Monitoring Diagram.drawio.png (image/png) Monitoring Diagram.drawio.png (image/png) Monitoring Diagram.drawio.png (image/png)