Postmortem Oficial

1. Visão Geral

1.1 Resumo do Incidente

Título do Incidente: Problemas de Desempenho e Instabilidade na API de Transações Financeiras

Sistemas Afetados: Aplicação AppPsSre (API).

Impacto: A aplicação estava lenta e instável, com falhas de conexão, o que deixou os usuários insatisfeitos.

---

2. Análise da Causa Raiz

2.1 O que Aconteceu?

A aplicação estava enfrentando sérios problemas de desempenho. Os usuários reclamavam que ela estava muito lenta e falhando nas conexões. A infraestrutura original tinha algumas limitações:

- A aplicação rodava em um único pod dentro de um cluster K3S.
- O banco de dados PostgreSQL era acessado diretamente por esse pod.
- O tráfego de rede era gerenciado por um Ingress Controller simples.

Essas configurações não eram suficientes para lidar com o aumento de carga e garantir a estabilidade.

2.2 Por que Aconteceu?

Minha análise mostrou os seguintes problemas principais:

- **Falta de Escalabilidade:** Só havia um pod para rodar a aplicação no K3S, o que criava gargalos quando a carga aumentava.
- **Falta de Redundância:** Não havia backup ou alternativas para o Ingress Controller e o banco de dados, o que significava que, se eles falhassem, a aplicação parava.
- **Problemas de Conexão:** As configurações para acessar o banco de dados não estavam otimizadas, o que resultava em falhas e lentidão.
- **Monitoramento Inadequado:** Não havia um sistema robusto para monitorar e analisar problemas em tempo real, o que dificultava a identificação e a solução rápida dos problemas.

2.3 Lições Aprendidas

Entendemos que a principal causa dos problemas foi a falta de uma infraestrutura mais robusta e resiliente. Precisamos adotar uma abordagem mais estruturada, com práticas como automação e monitoramento contínuo, para garantir que a aplicação funcione de maneira estável.

---

3. Remediação e Melhorias

3.1 Ações Imediatas Tomadas

Para resolver os problemas rapidamente, tomei as seguintes medidas:

- **Migração do K3S para EKS:** A infraestrutura será migrada para o EKS da AWS, proporcionando uma solução mais escalável e resiliente.
- **Melhoria na Escalabilidade:** Configurei um Horizontal Pod Autoscaler (HPA) para adicionar ou remover pods conforme a carga da aplicação aumenta.

- **Ajustes de Conexão com o Banco de Dados:** Otimizar as configurações de conexão com o PostgreSQL, reduzindo as falhas e melhorando o desempenho.
- **Revisão de Consultas SQL:** Otimizei as consultas SQL para que o banco de dados responda mais rapidamente por Cache.

3.2 Melhorias a Longo Prazo

Baseado na nova arquitetura, implementei as seguintes melhorias para garantir um desempenho estável e resiliente:

- **Alta Disponibilidade e Redundância:**
  - **Replicação de Pods:** Adicionei múltiplos pods no EKS, com balanceamento de carga via Nginx Ingress Controller.
  - **Redundância do Ingress Controller:** Implementei réplicas do Ingress Controller para garantir que ele não falhe.
  - **Banco de Dados e Cache:** Configurei replicação e failover automático para o banco de dados PostgreSQL e o Redis Cache.
  - **Load Balancer Externo:** Adicionei um Load Balancer para gerenciar o tráfego de entrada, garantindo que ele esteja sempre disponível, mesmo que uma zona de disponibilidade falhe.
- **Monitoramento e Observabilidade:**
  - **Monitoramento Completo:** Implementei o Prometheus e Grafana para monitorar a aplicação em tempo real e configurar alertas para problemas críticos.
  - **Logging Centralizado:** Introduzi Fluentd, Elasticsearch e Kibana para centralizar e analisar logs, o que facilita a identificação de problemas.
- **Escalabilidade Automática:**
  - **Horizontal Pod Autoscaler (HPA):** Configurei o HPA para ajustar automaticamente o número de pods conforme a carga da aplicação aumenta.
- **Resiliência e Tolerância a Falhas:**
  - **Chaos Engineering:** Implementei testes de resiliência para simular falhas e garantir que a infraestrutura aguente bem esses problemas.
  - **Políticas de Reinício e Verificação de Saúde:** Configurei as verificações de saúde para garantir que falhas nos pods sejam detectadas e corrigidas automaticamente.

3.3 Itens de Ação

- **Responsável:** Time DevOps/SRE
- **Descrição:** Implementar as correções necessárias para melhorar a estabilidade e o desempenho da aplicação.
- **Data de Conclusão:** N/A
- **Status:** N/A

---

4. Acompanhamento e Próximos Passos

4.1 Monitoramento e Métricas

- **Escalabilidade:** Vamos monitorar continuamente o HPA para garantir que a aplicação escale conforme necessário.
- **Disponibilidade:** Vamos monitorar as réplicas do Ingress Controller e os serviços de banco de dados para garantir que estejam sempre disponíveis.
- **Load Balancer:** Vamos monitorar o desempenho e o failover do Load Balancer para garantir que ele funcione bem em diferentes zonas de disponibilidade.
- **Desempenho:** Vamos avaliar continuamente as latências de conexão e a resposta da aplicação e do banco de dados.

4.2 Plano de Comunicação

- **Atualizações Periódicas:** Faremos reuniões mensais para revisar o desempenho da infraestrutura e discutir melhorias.

4.3 Documentação

- **Guias de Operação:** Atualizamos todos os guias de operação e resposta a incidentes com as novas configurações e procedimentos.

- **Documentação de Postmortem:** Este postmortem será mantido como referência para futuros incidentes e revisões de arquitetura.

4.4 Medidas Preventivas Futuras

- **Avaliações Regulares de Arquitetura:** Realizaremos revisões trimestrais da arquitetura para garantir que a infraestrutura continue a atender às necessidades de crescimento e resiliência.
- **Treinamento em Chaos Engineering:** Nossa equipe passará por treinamentos regulares em Chaos Engineering para melhorar nossa capacidade de responder a falhas inesperadas.

---

5. Conclusão

Este incidente mostrou que nossa arquitetura original tinha algumas vulnerabilidades, especialmente em termos de escalabilidade e resiliência. Conseguimos identificar as causas dos problemas e implementar melhorias significativas. As ações imediatas, como a otimização das consultas SQL e a configuração do HPA, trouxeram resultados rápidos. As melhorias de longo prazo, como a implementação do Load Balancer Externo e a configuração de monitoramento e observabilidade, garantem que estamos melhor preparados para lidar com futuros desafios.