# SRE

## Site Reliability Engineering

# Agenda

| |
|---|
| SRE além do cargo; |
| Observabilidade; |
| Service Levels; |
| Práticas de Resiliências e Recuperação; |
| PRR; |
| On-call, war room, incidentes. |

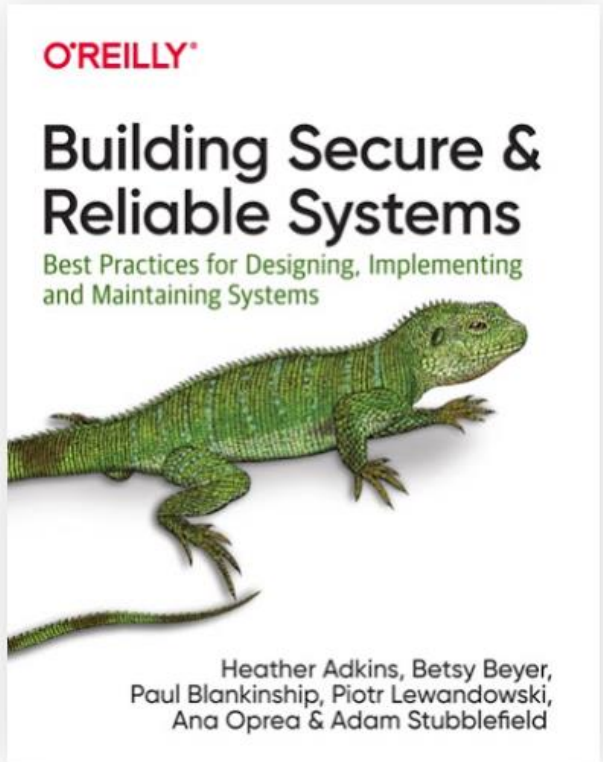
# Dinâmica do Workshop

- Vamos focar nos fundamentos!
- Provocações para aplicar no dia a dia!

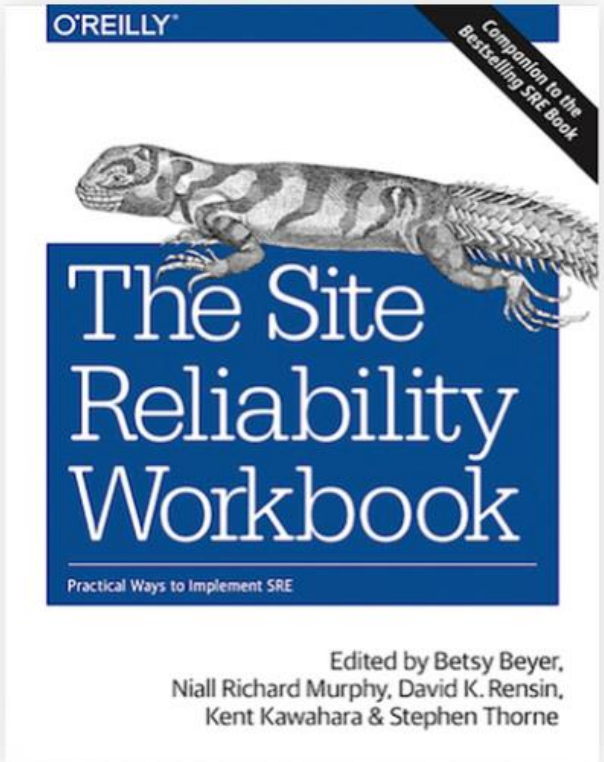
# SRE ou Site Reliability Engineering

- Google
- Foco em Aumentar e confiabilidade, disponibilidade
- Engenharia como Ops

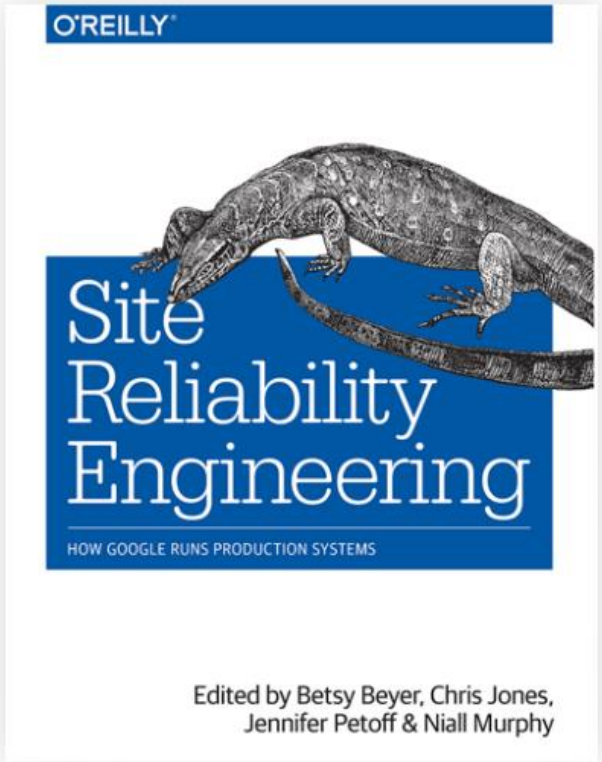
# SRE Books




Read online View details




Read online View details




Book updates

Read online View details

# Skill do Cargo

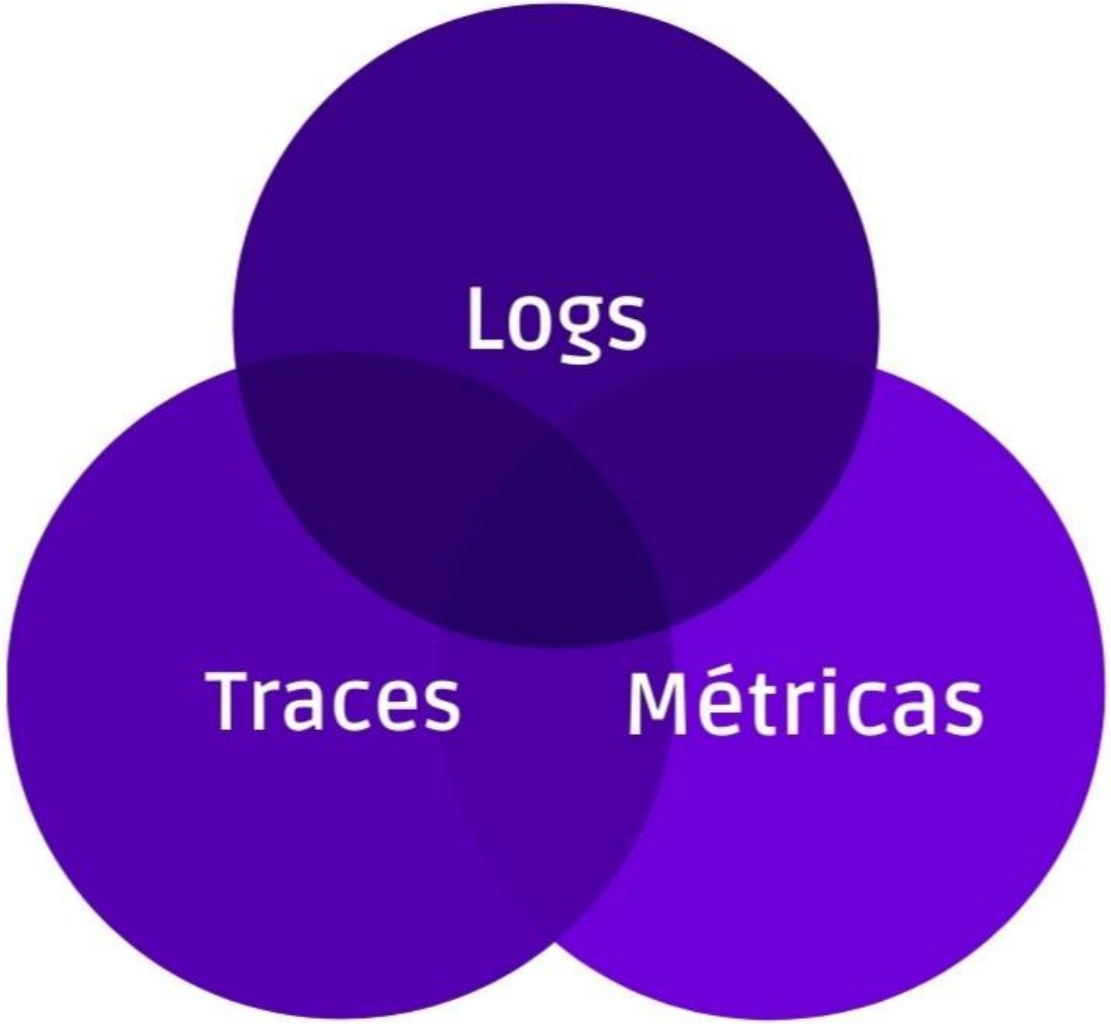
| |
|---|
| Habilidades em Programação e Automação |
| Forte Orientação a Confiabilidade e Resiliência |
| Experiência em Infraestrutura como Código (IaC) |
| Trabalho em Equipe e Comunicação |
| Monitoramento e Observabilidade |
| Orquestração e Gerenciamento de Containers |
| Pipeline de CI/CD |
| Gestão de Incidentes e Resposta a Problemas |
| Experiência em Cloud Provider |

# SRE como prática de Engenharia!

| |
|---|
| Aplique conceitos de observabilidade nas suas aplicações; |
| Defina Service Levels, garanta disponibilidade das suas aplicações; |
| Tenha qualidade como uma obrigação, tenha uma boa cobertura de testes, testes integrados, faça testes de estresses, carga, chãos etc; |
| Implemente práticas de tolerância a falhas, Resiliência e Recuperação; |
| Tenha rollbacks rápidos caso aconteça uma falha; |
| Desenvolva documentação, runbooks em caso de falhas/incidentes, entenda o que aconteceu e melhore, arrume o que não está bom para não acontecer novamente. |

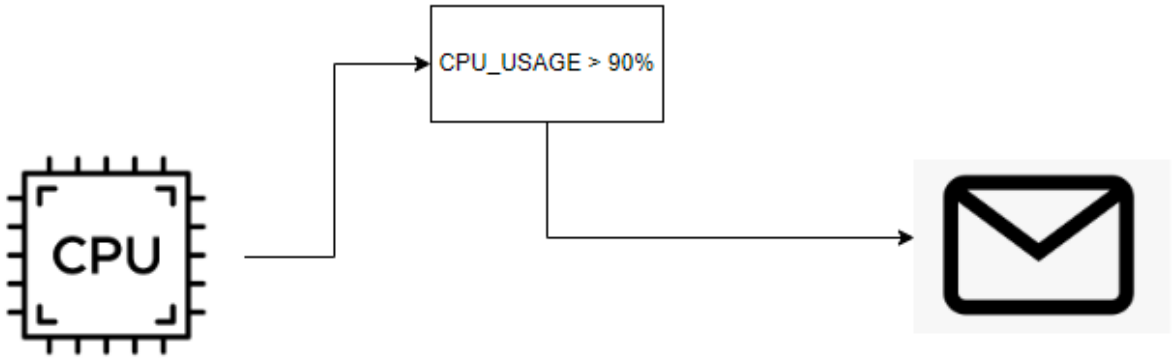
# Observabilidade

- Logs: Capturam Eventos em um determinado tempo.
- Métricas: Dados agregados ao longo do tempo;
- Tracing: Mostra o caminho de uma requisição.



# Monitoramento

- Sistemas vão quebrar, por isso precisamos ter técnicas para sermos alertados antes que isso aconteça!
- Utilizamos métricas e threshoulds (limites máximos/mínimos) para detectarmos e acionarmos uma ação.

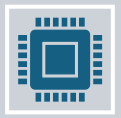


# 4 golden signs

Latência: O tempo que um serviço demora para completar uma requisição. **Importante para métricas**: requests com status de erros como por exemplo 5xx não devem ser contabilizadas nas métricas de latência.

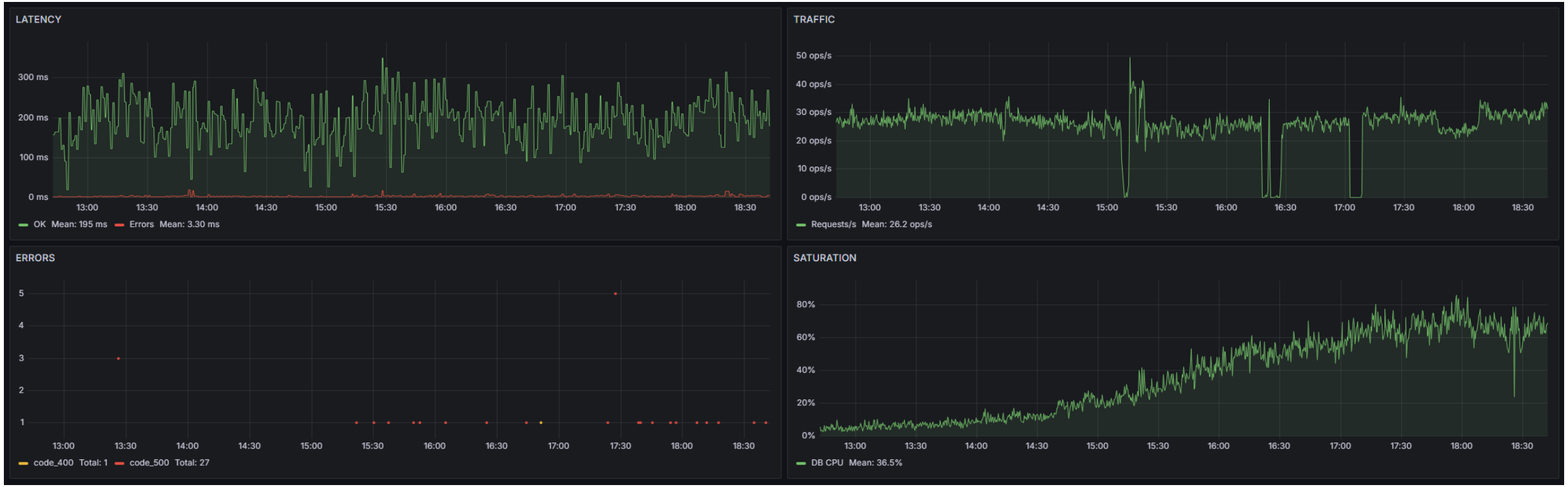
Tráfego: O quanto está sendo demandado pelo seu sistema. Ex. Request per Second (RPS) e Ios

Errors: Quantidade de erros (status code 5xx) ou algum outro retorno que pode trazer indisponibilidade ao serviço.



Saturação: Quanto meio serviço está cheio. Memoria, IO, Processamento.

# 4 Golden Sign Demonstrativo



# HomeWork

| |
|---|
| Eu conheço as principais dashboards dos produtos que eu atuo? |
| Consigo usar a observabilidade no meu dia a dia? |
| Minha squad tem 4 golden signs? |
| Minha squad tem alertas que são acionados antes que uma catástrofe aconteça? Threshoulds bem definidos? |

# Disponibilidade

- Tempo total que um serviço está disponível
- Podemos calcular de várias formas, uma forma simples é da seguinte maneira:

$$\text{Disponibilidade (\%)} = \left( \frac{\text{Tempo total de serviço} - \text{Tempo de indisponibilidade}}{\text{Tempo total de serviço}} \right) \times 100$$

# Service Levels

**Quando a gente a base da nossa observabilidade podemos começar a criar indicadores para nossas aplicações.**

**SLI (Service Level Indicator): apresenta a métrica de uma serviço em tempo real. Como latência, disponibilidade.**

**SLO (Service Level Objective): Objetivo que um serviço tem baseado no SLI.**

**SLA (Service Level Agreements): Garantir quanto de disponibilidade vai ser entregue em contrato e quais são as consequências caso isso for quebrado**

# Exemplo

Seu Chefe quer saber a disponibilidade dos últimos 30 dias do microsserviço de pagamento de boletos:

Você retorna para seu chefe que o SLI dos últimos 30 dias é de 99,98% de disponibilidade

Seu chefe quer saber qual foi o objetivo que o time acordou que deveria ser para esse microsserviço:

Voce retorna para seu chefe que o SLO acordado foi de 99,95%

O SLA em contrato é de 99,9% de disponibilidade, seu chefe passa para o time comercial que nos último 30 dias não tivemos nenhuma quebra de contrato!

https://uptime.is/

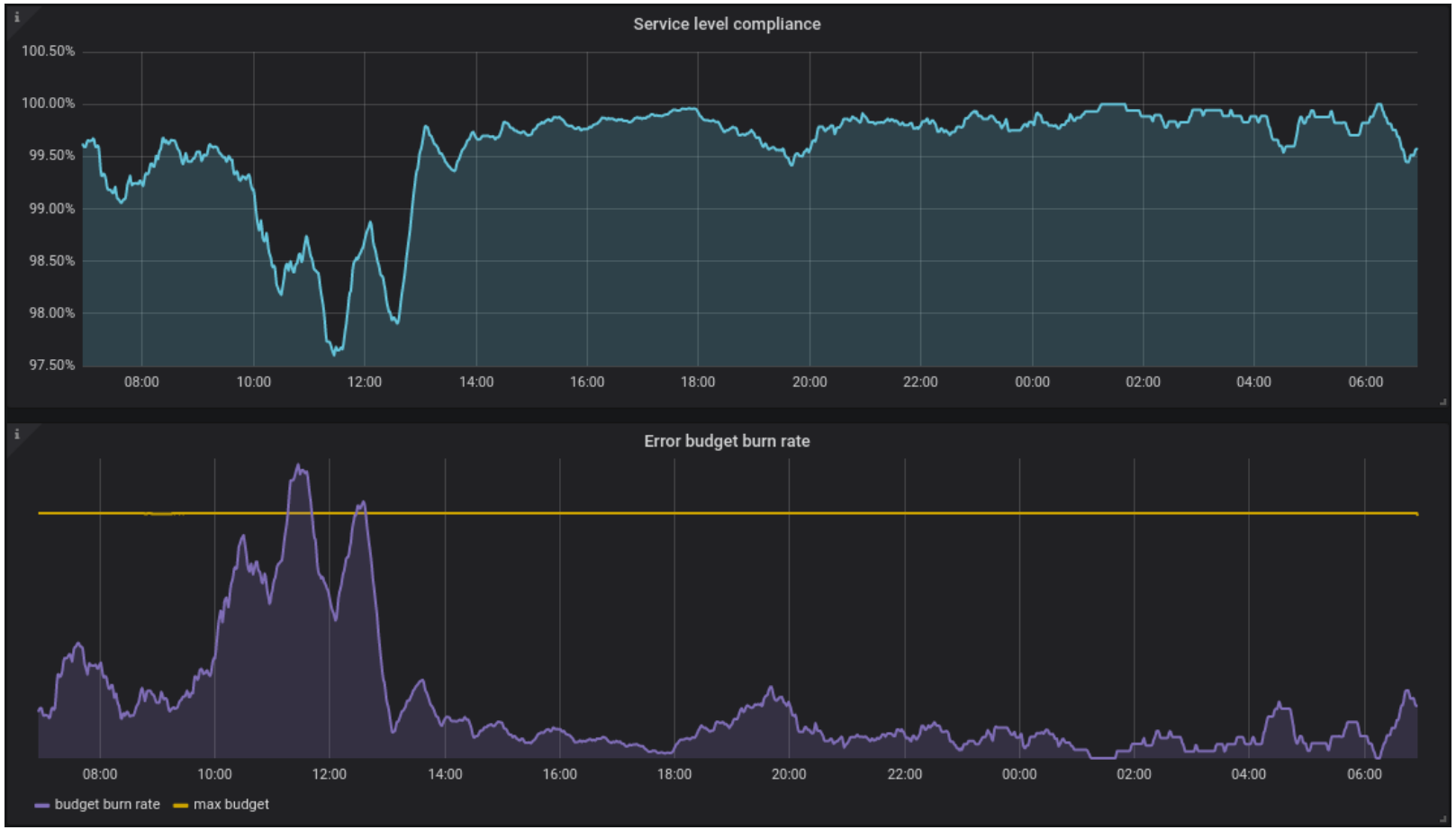
# Minha aplicação nunca vai poder falhar?

ela vai! e por isso existe o Error Budget...

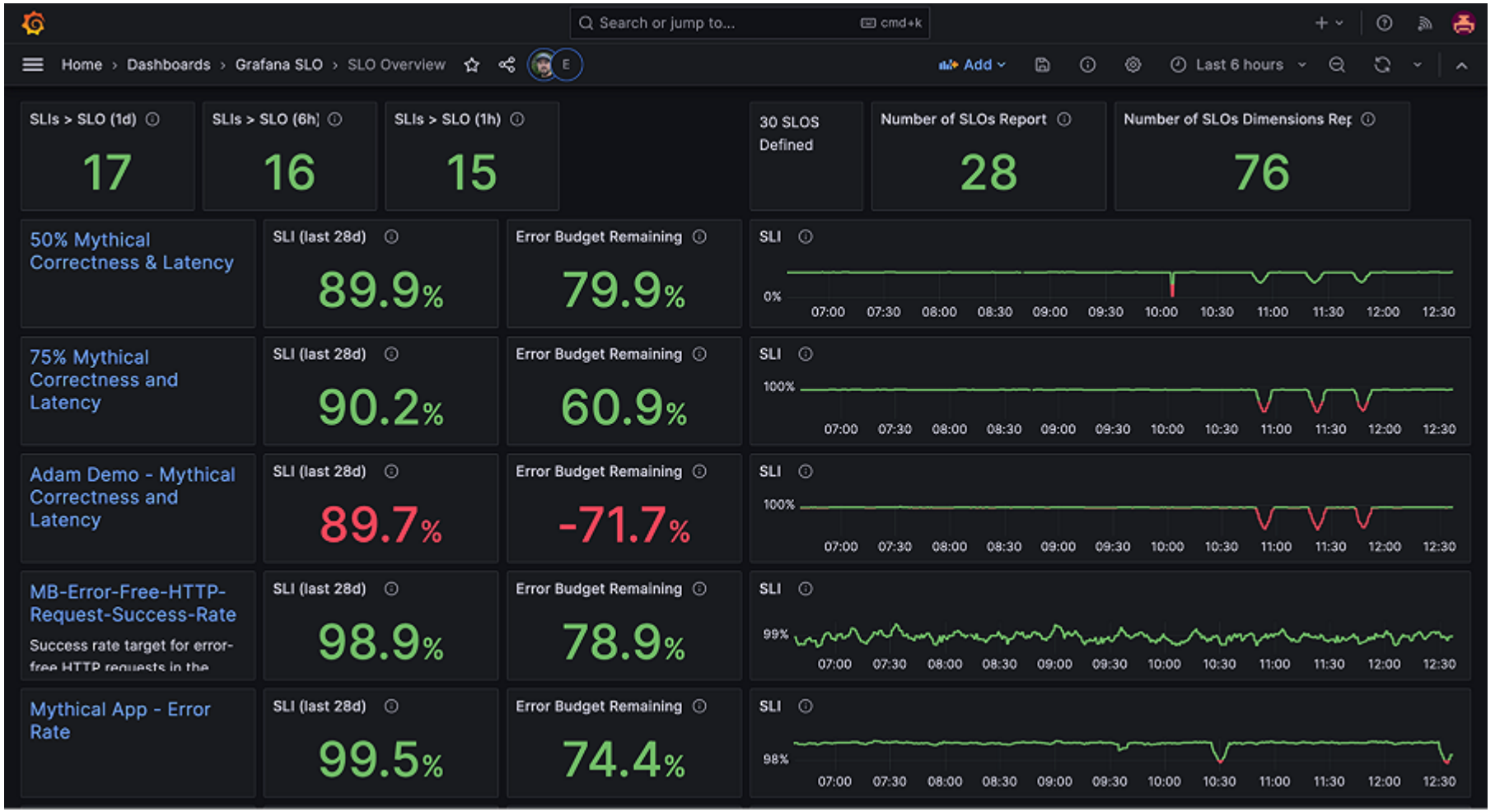
- Quantidade máxima de falhas/indisponibilidade que um serviço pode ter ao longo do tempo. Exemplo (janelas de manutenção previstas)
- Importante termos métricas e alarmes para o consumo do error budget.
- Recomendado termos práticas de rollback automatizado conforme formos consumindo o errorbudget durante um deploy com problemas em produção.

$$\text{Error Budget} = 100\% - \text{SLO}$$

# Exemplo de Consumo de Error Budget no grafana



# Dashboard para acompanhar Service Levels



# HomeWork

- Meu time já definiu quais são os SLOs dos produtos que eu atuo?
- Consigo ter acesso aos SLIs mensais dos produtos que atuo?

# PRR (Production Readness Review)

Processo de avaliação que indica o quanto uma aplicação, está preparada ser lançada em ambiente produtivo

Todos os esses pontos que conversamos e outros são avaliados junto com os tech leads e especialistas;

Ciclo de melhoria continua.

# Incidentes, On-Call e War Room

Por mais que criamos técnicas que processos para termos segurança, qualidade e resiliência em sistemas, incidentes vão acontecer.

Quem deveria cuidar do produto? O Desenvolvedor? O SRE? O Time de Operação? *You built it, you maintain it*

# On-Call

Chamamos de On-Call mas também é o mesmo que plantão, sobreaviso.

Geralmente é uma escala de 1 pessoa por dia, que fica de sobre aviso, caso aconteça alguma alerta, essa pessoa deverá entrar e entender o que está acontecendo.

# On-Call

## Níveis de maturidade de um time que faz oncall:

## Nível excelente:

- Escala organizada;
- Uma pessoa por dia;
- Follow de Sun / Turnos;
- Para cada alerta existe um runbook (documento ou automação que diz o que deve ser feito caso aconteça o alerta)
- Threshoulds bem definidos para alertas;
- Práticas de Postmortens e reports;
- Auto recuperação em caso de desastres;

# On-Call

## Níveis de maturidade de um time que faz oncall:

## Nível em desenvolvimento:

- A pessoa pode ficar mais de um dia no oncall;
- 24 horas ou mais de oncall;
- Muitas horas de interjornadas;
- Poucos ou não existes runbooks para alertas;
- Alert Fatigue a Falsos positivos; (Muitos alertas);
- Time não faz report e não tem métricas;
- Pouca ou nenhuma automação

# On-Call

## Níveis de maturidade de um time que faz oncall:

## Nível em Insuficiente:

- Não tem escala definida geralmente o gestor liga para quem ele consegue contato
- Pouco ou alertas insuficientes, o cliente ou outro responsável de outra área avisa do problema
- Time que não construiu o sistema está mantendo (time de operação)
- Métricas inexistentes

# Home Work

Hoje eu faço On-call? Minha squad se enquadra em qual nível de maturidade? Quais desses itens precisamos melhorar? Consigo pesquisar e criar um plano de ação?

Caso eu não faça On-call e meu tech lead / gestor me peça para começar no próximo mês eu estou preparado? Avalie o nível de maturidade do On Call do seu projeto, de ideias, seja pro ativo!

# Durante a Tempestade

...

A temida war room!

Mas formalmente chamamos de Sala de Incidentes.

Os papeis!

- Incident Comander: líder que vai tomar as decisões durante a crise
- Engenheiro do sistema ou Operador do Sistema: o que vai identificar o problema e fazer o troubleshooting
- Observador: anota e documenta as ações e decisões tomadas durante o incidente para serem usadas posteriormente;
- Comunicador: responsável por comunicar os interessados sobre o incidente / problema.

# Simulação de Incidente

# Postmortem

# Práticas de Resiliências e Recuperação

**Rollback (retorno de pacote após implantação)**

Implemetei em produção achei um bug devo arrumar e mandar para produção?

- Somente em último caso... O certo é fazer o rollback do pacote da implantação;

Alterações menores e mais seguidas;

Alterações que possam ser reversíveis;

Tenha um plano de Retorno descritivo com passos (outra pessoa poderá ter que implementar ou ter que reverter o pacote);

Garanta que nenhuma métrica foi alterada e nenhum alarme foi desativad após a alteração;

Caso tenha feito o rollback, faça um POSTMORTEM

Se for uma implementação para fluxo crítico, é possível desenvolver uma feature flag?

**Feature Flag (Permite ativar/desativar funcionalidades especificas sem a necessidade de uma nova implementação)**

É uma boa estratégia para testar implantações críticas ou testar funcionalidades para não ferir o indicador de error budget e disponibilidade;

**HotFix (Resolução de problemas em produção fora do fluxo normal de desenvolvimento – dev, hom)**

Patches de segurança críticos;

Erros Críticos;

Quebra de contratos com integrações.