INSTITUTO BRASILEIRO DE ENSINO, DESENVOLVIMENTO E PESQUISA - IDP
CURSO: Ciência da Computação / Engenharia de Software

DISCIPLINA: Redes de Computadores e Internet

PROFESSORA: Lorena Borges

TÍTULO: Plataforma de Monitoramento para DevOps: Arquitetura, Implementação e Resposta a Incidentes

ALUNOS: Renato Portilho Costa e Yuri Buenos Aires Santana

Brasília - DF Dezembro/2025

1. INTRODUÇÃO E OBJETIVO

Este projeto tem como objetivo projetar e implementar uma plataforma de monitoramento de serviços de rede focada em práticas DevOps. A solução visa garantir a alta disponibilidade, desempenho e segurança de serviços críticos (Web, Banco de Dados, DNS e SMTP), fornecendo visibilidade em tempo real através de dashboards e um sistema de alerta proativo para resposta a incidentes.

O sistema foi desenvolvido para atender aos requisitos de monitoramento de infraestrutura, ingestão de métricas e detecção de anomalias de segurança, conforme especificado no escopo do trabalho final da disciplina.

2. ARQUITETURA DA SOLUÇÃO

A plataforma foi desenvolvida utilizando uma arquitetura modular baseada em microsserviços, dividida em quatro camadas principais:

1. **Agente Coletor (Python):** Script autônomo responsável pela sondagem ativa dos serviços. Ele executa testes de disponibilidade (HTTP/ICMP), resolução de nomes (DNS), handshake de e-mail (SMTP) e integridade de arquivos (FIM).
2. **Backend de Ingestão (API Flask):** Uma API REST que recebe os dados coletados, aplica regras de negócio (Classificação de Risco Níveis 1, 2 e 3) e gerencia a lógica de notificação via e-mail real (SMTP Gmail).
3. **Armazenamento (MongoDB/Docker):** Banco de dados NoSQL utilizado para persistir o histórico de métricas e logs de incidentes, rodando em container para isolamento e facilidade de deploy.
4. **Frontend (Dashboard HTML5/JS):** Interface gráfica responsiva com tema "Dark Mode", desenvolvida com Bootstrap e Chart.js, consumindo a API para renderização de gráficos em tempo real.

3. MONITORAMENTO IMPLEMENTADO

A solução cobre as métricas exigidas para os seguintes serviços:

- **Web Server:** Monitoramento de disponibilidade (Status Code), latência (ms) e detecção de quedas (HTTP 500/404).
- **Banco de Dados:** Monitoramento de status (UP/DOWN), tamanho em disco, QPS (Queries per Second) e Slow Queries simuladas para validação de dashboard.
- **DNS:** Tempo de resposta na resolução de domínios e taxa de erros de consulta.
- **SMTP:** Latência de conexão, tamanho da fila de envio e taxa de entrega/erro.
- **Segurança (Extras):** Detecção de anomalias de tráfego (DDoS), tentativas de Brute-Force e alteração de arquivos de configuração (File Integrity Monitoring).

4. GUIA DE INSTALAÇÃO

Para a execução do ambiente, são necessários: Docker e Python 3.9+.

Passo 1: Inicialização do Banco de Dados Executar o container do MongoDB: `docker run -d -p 27017:27017 --name mongo-devops mongo:latest`

Passo 2: Execução do Backend (API) Instalar dependências e iniciar o servidor Flask: `pip install flask pymongo flask-cors python backend/app.py` *O sistema iniciará na porta 5000 e validará as credenciais de envio de e-mail.*

Passo 3: Execução do Agente de Monitoramento Iniciar o script de coleta em um novo terminal: `pip install requests psutil dnspython python agente/coletor.py`

Passo 4: Acesso ao Dashboard Abrir o arquivo `frontend/index.html` em qualquer navegador web moderno.

5. RUNBOOKS E PLAYBOOKS DE INCIDENTES

Conforme exigido na entrega, abaixo estão detalhados os procedimentos operacionais padrão.

5.1. Regras de Alerta (SLA)

O sistema classifica incidentes em três níveis:

- **Nível 1 (Verde):** Operação normal. Nenhuma ação necessária.
- **Nível 2 (Amarelo):** Degradação de performance (ex: Latência > 1s, CPU > 75%). Ação: Notificação de alerta por e-mail.
- **Nível 3 (Vermelho):** Indisponibilidade ou Falha Crítica. Ação: Notificação imediata e escalonamento.

5.2.

Playbook A: Resposta a Ataque DDoS

- **Gatilho:** O sistema detecta > 50 requisições em uma janela de 10 segundos.

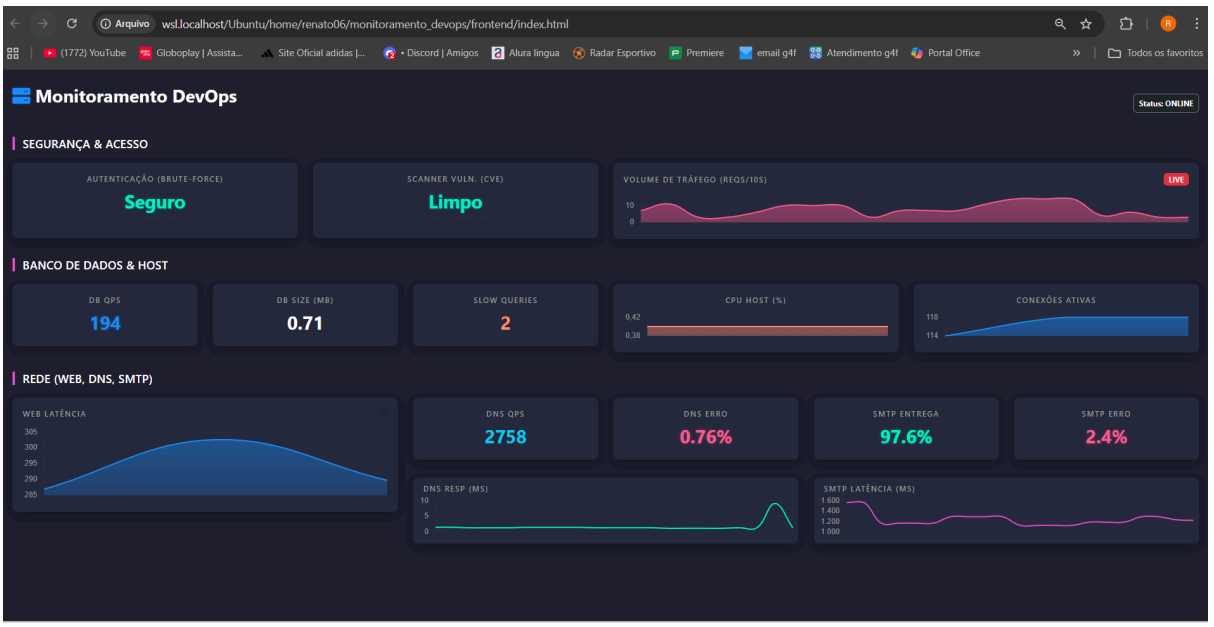
- **Sintoma:** Dashboard exibe alerta visual "TRÁFEGO ANÔMALO" e e-mail crítico é enviado.
- **Procedimento Automático:** O Backend registra o IP e bloqueia o processamento da métrica.
- **Ação do Operador:**
    1. Verificar logs de rede no Firewall de borda.
    2. Identificar IPs de origem do ataque.
    3. Aplicar regra de *drop* no WAF (Web Application Firewall).

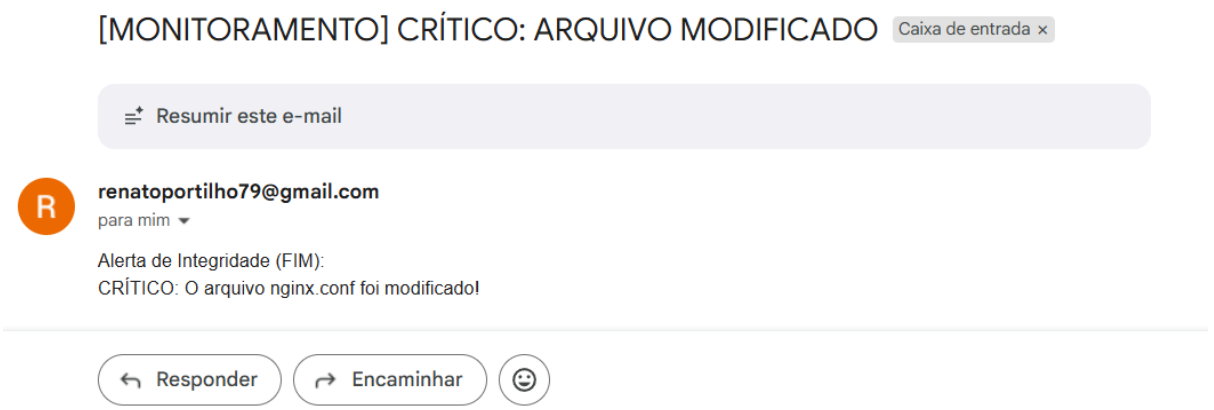
5.3.

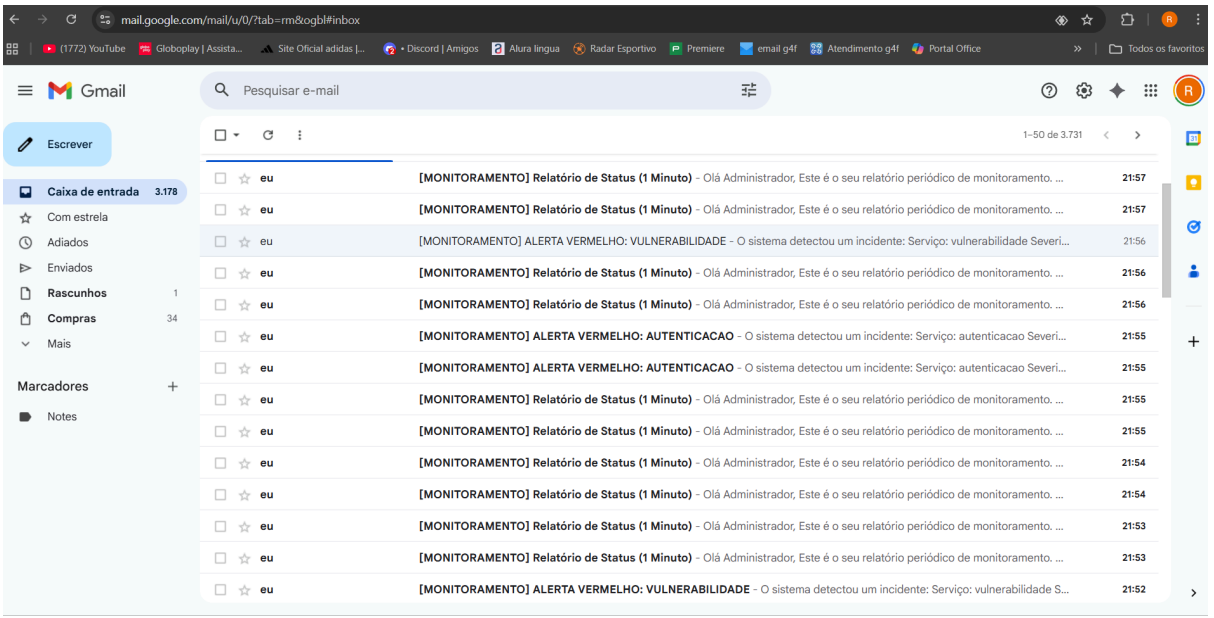
Playbook B: Violação de Integridade (Configuração)

- **Gatilho:** Alteração no hash SHA-256 do arquivo `nginx.conf`.
- **Sintoma:** E-mail enviado com assunto "CRÍTICO: ARQUIVO MODIFICADO".
- **Ação do Operador:**
    1. Acessar o servidor via SSH imediatamente.
    2. Verificar o histórico de comandos (`history`) para identificar o autor.
    3. Se a alteração não foi planejada (Change Management), restaurar o backup da configuração anterior.
    4. Rotacionar credenciais de acesso ao servidor.

6. EVIDÊNCIAS DE TESTES







7. CONCLUSÃO

O projeto atendeu a todos os requisitos propostos, implementando um ciclo completo de monitoramento DevOps. A solução demonstrou capacidade de detectar falhas de disponibilidade e incidentes de segurança em tempo real, notificando os administradores via e-mail e permitindo uma resposta rápida através da visualização centralizada no Dashboard.