Documentação do Sistema de Monitoramento Automatizado de Servidor Nginx - AWS

Este projeto documenta a criação de um sistema automatizado para monitoramento do estado de um servidor Nginx em dois ambientes: uma máquina local usando o Windows Subsystem for Linux (WSL) com Ubuntu e uma instância Ubuntu hospedada na AWS. O sistema utiliza scripts e agendamento via cron para garantir monitoramento periódico, com registros armazenados para auditoria.

Índice

Objetivos do Projeto

- Configurar um subsistema Ubuntu no WSL ou uma instância Ubuntu na AWS.
- Implementar o servidor Nginx como ambiente de teste.
- Desenvolver um script que valida o status do Nginx.
- Registrar logs com informações detalhadas sobre a validação.
- Automatizar a execução do script para rodar a cada 5 minutos.

Requisitos

1. Windows 10 ou superior: Com suporte ao WSL (Windows Subsystem for Linux).
2. Ubuntu 20.04 ou superior: Disponível no WSL ou em uma instância EC2 na AWS.
3. AWS CLI: Instalado e configurado no sistema.
4. Conta AWS: Para configurar os recursos no AWS Console.
5. Servidor Nginx: Instalado no ambiente configurado.

---

1. Configuração do Ambiente

- Configuração do WSL com Ubuntu no Windows

1.1. Instale o WSL:

wsl --install

- Reinicie o computador após a instalação.

1.2. Instale o Ubuntu:

- Baixe o Ubuntu 20.04 ou superior da Microsoft Store.

1.3. Atualize os pacotes no WSL:

sudo apt update && sudo apt upgrade -y

2. Instalação do AWS CLI no WSL

1. Baixe e instale o AWS CLI:

curl"https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

sudo apt install unzip -y

unzip awscliv2.zip

sudo ./aws/install

2. Verifique a instalação:

aws --version

3. Configure o AWS CLI:

aws configure

Forneça:

- Access Key ID e Secret Access Key.
- Região padrão (ex.: us-east-1).
- Formato de saída padrão (ex.: JSON).

3. Configuração de Infraestrutura na AWS

Criar uma VPC

1. No console da AWS, acesse VPC > Your VPCs > Create VPC.
2. Configure:
   - Name tag: Nome da VPC (ex.: `MyVPC`).
   - IPv4 CIDR block: `10.0.0.0/16`.
3. Clique em Create VPC.

Criar uma Sub-rede Pública

1. No console da VPC, acesse Subnets > Create Subnet.
2. Configure:
   - Name tag: Nome da sub-rede (ex.: `MyPublicSubnet`).
   - VPC: Selecione a VPC criada.
   - IPv4 CIDR block: `10.0.1.0/24`.
3. Clique em Create Subnet.

Configurar um Gateway de Internet

1. Em Internet Gateways > Create Internet Gateway.
2. Configure:
   - Name tag: Nome do gateway (ex.: `MyInternetGateway`).
3. Clique em Create Internet Gateway.
4. Selecione o gateway, clique em Attach to VPC e escolha sua VPC.

Configurar a Tabela de Rotas

1. Em Route Tables > Create Route Table.
2. Configure:
   - Name tag: Nome da tabela (ex.: `MyRouteTable`).
   - VPC: Selecione sua VPC.
3. Adicione uma rota:
   - Destination: `0.0.0.0/0`.
   - Target: Selecione o gateway de internet.
4. Associe a tabela à sub-rede pública.

Configurar um Security Group

1. Em Security Groups > Create Security Group.
2. Configure:
   -Name tag: Nome do grupo (ex.: `MySecurityGroup`).
   - Description: Permitir acesso SSH e HTTP.
3. Adicione regras:
   - Type: `SSH`, Port Range: `22`, Source: `My IP`.
   - Type: `HTTP`, Port Range: `80`, Source: `0.0.0.0/0`.

Configuração de Instância EC2

1. No console da AWS, acesse EC2 > Launch Instance.
2. Configure:

- AMI: Ubuntu 20.04 LTS.
- Instance Type: t2.micro.
- Key Pair: Crie ou selecione um par de chaves.
- Network Settings: Use as configurações criadas anteriormente.

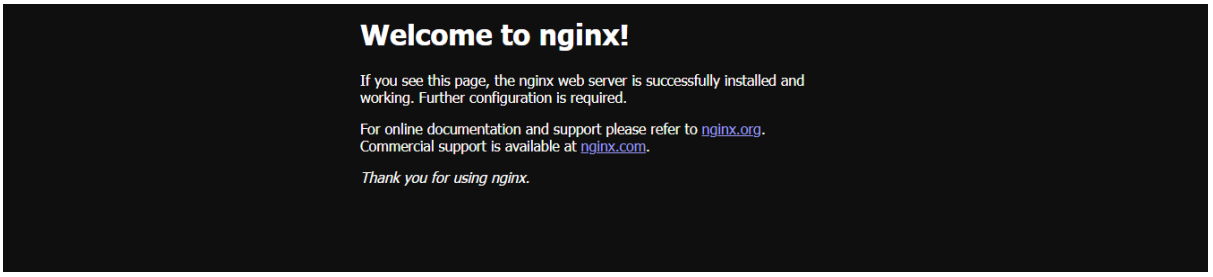
3. Conecte-se à instância via SSH:
   ssh -i chave.pem ubuntu@<ENDEREÇO_IP>

4. Instale e configure o Nginx:
   sudo apt update -y
   sudo apt install nginx -y
   sudo systemctl start nginx
   sudo systemctl enable nginx

---

4. Script de Monitoramento do Nginx

- Verifique se está funcional em seu navegador por meio do IP ou localhost:

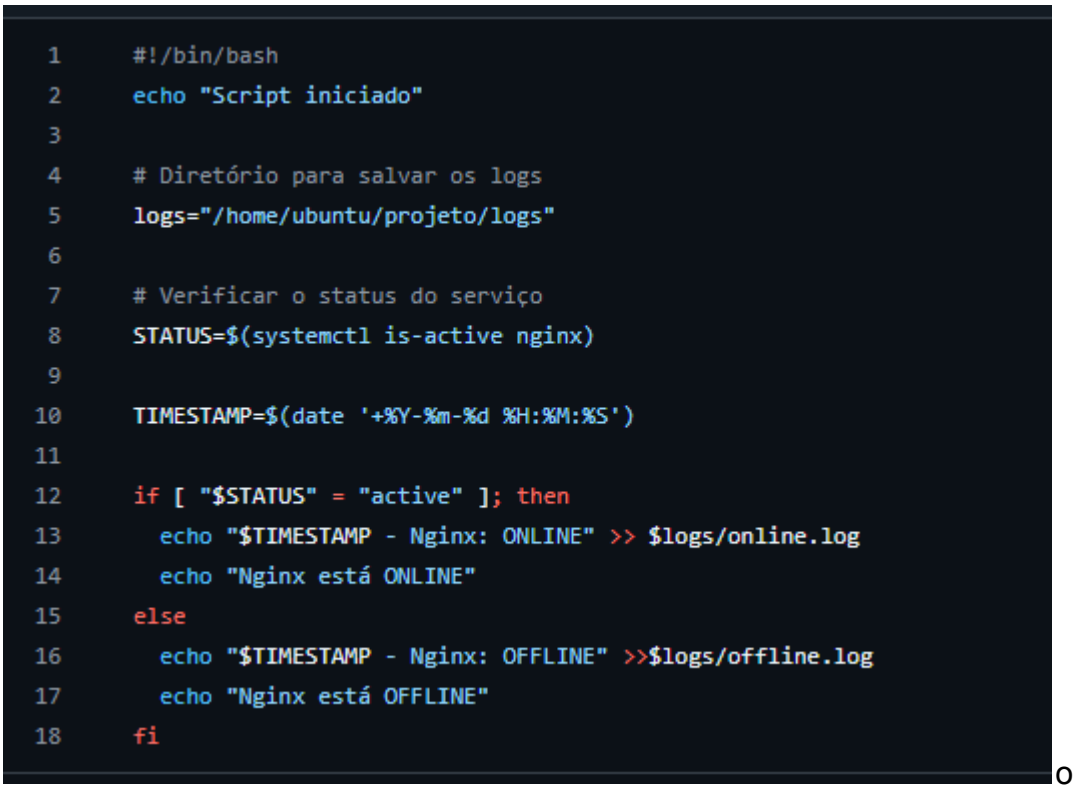


Antes de iniciar a criação do script, foi configurada uma estrutura organizada de diretórios para armazenar os arquivos do projeto.

1. crie um diretório chamado projeto:
   mkdir ~/projeto
2. Dentro de projeto, crie um subdiretório chamado logs para armazenar os arquivos de saída:
   mkdir ~/projeto/logs
3. Crie o arquivo do script chamado script_nginx.sh dentro do diretório projeto:

   touch ~/projeto/script_nginx.sh

4. Adicione o seguinte conteúdo



```
#!/bin/bash
echo "Script iniciado"

# Diretório para salvar os logs
logs="/home/ubuntu/projeto/logs"

# Verificar o status do serviço
STATUS=$(systemctl is-active nginx)

TIMESTAMP=$(date '+%Y-%m-%d %H:%M:%S')

if [ "$STATUS" = "active" ]; then
  echo "$TIMESTAMP - Nginx: ONLINE" >> $logs/online.log
  echo "Nginx está ONLINE"
else
  echo "$TIMESTAMP - Nginx: OFFLINE" >>$logs/offline.log
  echo "Nginx está OFFLINE"
fi
```

o:

- Salve e saia do editor (Ctrl+O, Enter, Ctrl+X).

5. Torne o script executável:

chmod +x ~/projeto/script_nginx.sh

---

5. Automatização com Cron

Para executar o script automaticamente a cada 5 minutos, utilize o cron.

1. Edite o crontab:
2. **sudo crontab -e**

Se for a primeira vez que você está abrindo o crontab, será solicitado que escolha um editor de texto. Recomendo o nano por ser mais simples.

● O comando crontab no Linux é um serviço de agendamento de tarefas automáticas para os usuários e o sistema. Ele permite que um comando, programa ou script seja agendado para um determinado dia, mês, ano e hora. É muito usado em tarefas que precisam ser executadas a cada hora, dia ou qualquer outro período, de forma recorrente.

A sintaxe :

```
* * * * * comando_a_ser_executado
| | | | |
| | | | +--- Dia da semana (0 - 7) (Domingo = 0 ou 7)
| | | +----- Mês (1 - 12)
| | +------- Dia do mês (1 - 31)
| +--------- Hora (0 - 23)
+----------- Minuto (0 - 59)
```

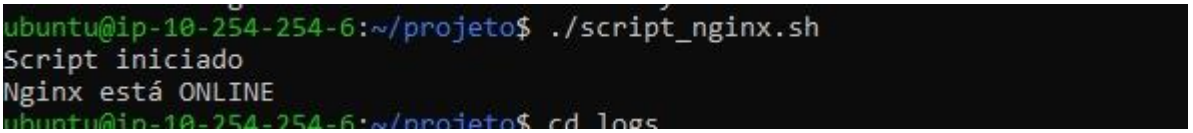
2. No editor do crontab, role até o final e adicione a seguinte linha:

   ***/5 * * * * ~/projeto/script_nginx.sh**

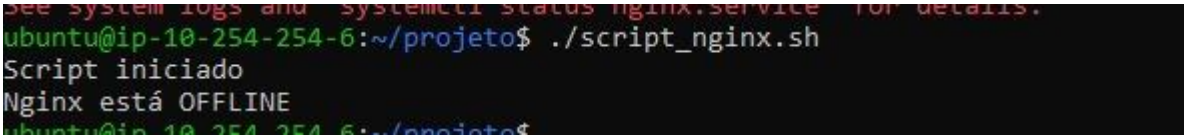
- No editor nano, salve o arquivo Crt + o, e feche-o, Crt + x.

● Com isso, o script será executado automaticamente a cada 5 minutos, gerando dois arquivos de saída no diretório logs:

online.log: Contém registros quando o serviço está ativo.



offline.log: Contém registros quando o serviço está inativo.



4. Verifique as tarefas agendadas:
   crontab -l

- Isso mostrará as tarefas agendadas. Você deve ver a linha que adicionou.

6. Testes e Validação

1. Verifique os arquivos de log:
   ls ~/projeto/logs

● O arquivo online.log será gerado quando o serviço Nginx estiver online.

● O arquivo offline.log será gerado caso o serviço Nginx esteja offline. Confirme que os registros estão sendo gerados corretamente.

2. Simule cenários para validação

● Teste o status online: Certifique-se de que o Nginx está rodando:

sudo systemctl start nginx

Aguarde 5 minutos e confira o conteúdo de online.log:

cat online.log

```
ubuntu@ip-10-254-254-6:~/projeto$ cd logs
ubuntu@ip-10-254-254-6:~/projeto/logs$ cat online.log
2025-01-19 08:58:20 - Nginx: ONLINE
2025-01-19 09:00:01 - Nginx: ONLINE
2025-01-19 09:05:01 - Nginx: ONLINE
2025-01-19 06:08:24 - Nginx: ONLINE
2025-01-19 06:10:01 - Nginx: ONLINE
2025-01-20 20:20:01 - Nginx: ONLINE
2025-01-20 20:21:08 - Nginx: ONLINE
```

- Teste o status offline: Pare o serviço do Nginx:
  sudo systemctl stop nginx

Após 5 minutos, verifique o conteúdo de offline.log:
cat offline.log

```
ubuntu@ip-10-254-254-6:~/projeto/logs$ cat offline.log
2025-01-19 06:15:01 - Nginx: OFFLINE
2025-01-19 06:15:37 - Nginx: OFFLINE
ubuntu@ip-10-254-254-6:~/projeto/logs$
```

Versionamento do Projeto

- Este documento descreve o processo utilizado para configurar o versionamento do projeto utilizando **Git** e **GitHub**.

Passos realizados:

1. Inicialização do repositório Git

No diretório raiz do projeto, o repositório Git foi inicializado com o seguinte comando:

git init

2. Configuração do repositório remoto

O repositório remoto foi configurado para sincronizar o projeto com o GitHub. O endereço remoto foi adicionado com o comando:

git remote add origin git@github.com:mariakarolina/Project---DevSecOps.git

3. Adicionando os arquivos para versionamento

Os arquivos do projeto foram adicionados ao estágio do Git:

git add .

4. Criando o commit inicial

Um commit inicial foi criado para registrar as primeiras mudanças no repositório:

git commit -m "Initial commit"

5. Configuração e carregamento da chave SSH

Para garantir a autenticação segura com o GitHub, uma chave SSH foi configurada:

- Verificação e carregamento da chave SSH no agente:

eval "$(ssh-agent -s)"

ssh-add ~/.ssh/id_ed25519

- Teste da conexão SSH com o GitHub:

ssh -T git@github.com

6. Envio do projeto para o GitHub

O projeto foi enviado ao repositório remoto no GitHub:

git push -u origin main

7. Sincronização do repositório remoto

Para garantir que o repositório local estivesse sincronizado com o remoto, foi utilizado:

git pull

Resultados Esperados

● Um ambiente Linux funcional no WSL ou na AWS
● Um servidor Nginx em execução.
● Um script que valida o status do serviço e registra logs de forma automatizada.
● Automação configurada via cron para garantir a execução periódica.