Self-Healing Infrastructure with Prometheus, Alertmanager & Ansible

Abstract:

This project demonstrates the implementation of a self-healing infrastructure using Prometheus, Alertmanager, and Ansible. The objective is to automatically detect service failures and recover them without manual intervention. By integrating monitoring, alerting, and automation, the system ensures high availability and minimizes downtime.

Introduction:

In modern DevOps practices, system reliability and uptime are crucial. Traditional monitoring systems only notify engineers of failures, requiring manual recovery. To improve efficiency, we built a self-healing infrastructure that automatically detects failures, triggers alerts, and recovers services seamlessly using Prometheus, Alertmanager, and Ansible.

Tools Used:

- **Prometheus** – Monitors service health and collects metrics.
- **Alertmanager** – Triggers alerts based on Prometheus rules.
- **Ansible** – Automates recovery actions via playbooks.
- **Shell Scripting** – Handles service checks and integrations.
- **Ubuntu VM/Docker** – Deployment and testing environment.

Steps Involved in Building the Project:

1. **Deploy Sample Service:** Set up NGINX on Ubuntu/Docker.
2. **Configure Prometheus:** Monitors NGINX uptime and system metrics.
3. **Set Alert Rules:** Trigger alerts for conditions like service downtime or CPU usage > 90%.
4. **Integrate Alertmanager:** Configure a webhook to invoke Ansible playbooks.
5. **Create Ansible Playbook:** Automates restarting failed services.
6. **Testing & Verification:** Stopped NGINX manually; auto-healing restarted it successfully.

Conclusion:

This project demonstrates a fully automated self-healing infrastructure, reducing manual intervention and improving system availability. By combining Prometheus, Alertmanager, and Ansible, we created a resilient monitoring and recovery mechanism suitable for modern DevOps pipelines.