**Self-Healing Infrastructure with Prometheus, Alertmanager & Ansible**

**Introduction**

In modern IT environments, service availability is critical for business continuity. Downtime can lead to loss of revenue and customer trust. Self-healing infrastructure automates the detection and recovery of service failures, reducing manual intervention. This project demonstrates a self-healing system using **Prometheus**, **Alertmanager**, and **Ansible** to monitor a service and automatically restore it upon failure.

**Abstract**

The objective of this project is to build an automated infrastructure that detects service disruptions and recovers without human intervention. A sample service (NGINX) is deployed and monitored using Prometheus. Alertmanager listens for failures and triggers an Ansible playbook to restart the service. This reduces downtime and improves system reliability.

**Tools Used**

* **Prometheus** – Monitors system metrics and service health.
* **Alertmanager** – Processes alerts from Prometheus and triggers automation.
* **Ansible** – Executes automated remediation actions.
* **Ubuntu VM/Docker** – Hosts the services.
* **NGINX** – Sample service for monitoring.
* **Shell Scripting** – Custom scripts for alert handling.

**Steps Involved**

**1. Deploy the Sample Service**

* Created an Ubuntu VM or Docker container.
* Installed NGINX.

sudo apt update && sudo apt install nginx -y

* Verified service availability at http://<IP>.

**2. Install & Configure Prometheus**

* Installed Prometheus on a monitoring server.
* Configured prometheus.yml to scrape metrics from the target server.

scrape\_configs:

- job\_name: 'nginx'

static\_configs:

- targets: ['<server-ip>:9113']

* Added exporters for NGINX metrics.

**3. Setup Alertmanager**

* Installed Alertmanager.
* Created an alert rule in Prometheus:

groups:

- name: service\_alerts

rules:

- alert: NginxDown

expr: up{job="nginx"} == 0

for: 1m

labels:

severity: critical

annotations:

summary: "NGINX service is down"

* Configured Alertmanager to trigger a webhook for automation.

**4. Ansible Playbook Creation**

* Created a playbook to restart the NGINX service:

- name: Restart NGINX service

hosts: all

become: yes

tasks:

- name: Restart nginx

service:

name: nginx

state: restarted

**5. Integrate Alertmanager with Ansible**

* Configured Alertmanager webhook to call a shell script that runs the Ansible playbook.

ansible-playbook restart\_nginx.yml -i hosts.ini

**6. Testing Auto-Healing**

* Stopped NGINX manually:

sudo systemctl stop nginx

* Verified that Prometheus detected the failure, Alertmanager triggered the Ansible playbook, and NGINX restarted automatically.

**Conclusion**

This project demonstrated an automated self-healing infrastructure using Prometheus, Alertmanager, and Ansible. It eliminates the need for manual intervention during service failures, improving uptime and operational efficiency. This setup can be extended to monitor and recover multiple services in production environments, providing robust fault tolerance.