

Informe Técnico: Monitoreo de Red

Fecha: 10 de julio de 2025

Autor: José Luis Sánchez Barrera, Ricardo Yael Pacheco Mendoza

Carrera: Tecnologías de la Información

Universidad: Universidad Tecnológica de Tula-Tepeji

***. Introducción**

El presente informe describe el desarrollo e implementación de un dashboard interactivo que permite el monitoreo de nodos dentro de una topología de red virtualizada. El proyecto se basa en herramientas de código abierto como Python, Flask, Netmiko, Chart.js y GNS3, con el objetivo de proporcionar una vista centralizada del estado y comportamiento de los dispositivos conectados.

***. Objetivos**

General

Desarrollar un monitoreo interactivo en Python con Flask para monitorear el estado, el rendimiento y la seguridad de los nodos de una red .

Específicos

- Integrar verificación de nodos mediante Ping, DNS y NTP.
- Visualizar métricas de estado, tráfico y topología en tiempo real.
- Ejecutar comandos remotos mediante SSH con Netmiko.
- Simular escaneos de puertos con Nmap.
- Gestionar y mostrar el historial de acciones.
- Presentar una interfaz web responsiva, funcional y clara.

***. Arquitectura del Sistema**

Diagrama General

- Nodo Principal: UbuntuDockerGuest-1 (ejecuta el dashboard y los scripts de monitoreo).
- Nodos monitoreados: Web-Docker-1, Web-Docker-2, DNS/NTP, Grafana.
- Router R3: enruta el tráfico entre subredes 10.0.0.0/24, 10.0.10.0/24, 10.0.20.0/24 y 10.0.40.0/24.

- Conectividad basada en GNS3.

***. Implementación Técnica**

Tecnologías

- Lenguaje: Python 3.12
- Framework Web: Flask 2.3
- Visualización: Chart.js (JS)
- SSH Automatizado: Netmiko
- DNS y NTP: Servicios Linux instalados en Docker
- Infraestructura: GNS3 con nodos Docker, switches y routers virtuales

Funcionalidades por Ruta

- / (Inicio): Menú de acceso
- /nodos: Verificación de conectividad mediante Ping
- /graficas: Gráficas de métricas por nodo (dinámicas)
- /trafico: Tabla de tráfico por nodo
- /topologia: Imagen de topología o generación automática
- /comandos: Ejecución controlada de comandos (ping, docker ps, etc.)
- /nmap: Simulación de escaneo Nmap por nodo
- /historial: Tabla de comandos ejecutados

5. Revisión del Código

- El proyecto está centralizado en un archivo principal (app.py)
- Plantillas HTML
- Migración de Matplotlib a Chart.js para eficiencia del lado cliente
- Escaping de salidas para prevenir XSS
- Comandos restringidos y controlados con seguridad básica

***. Revisión**

Funcionalidad	
/nodos	Simulación por ping con actualización cada 10 segundos
/graficas	Gráficas con Chart.js, datos simulados
/trafico	Tabla actualizable con datos simulados
/topologia	Imagen estática, falta generador dinámico
/comandos	Ejecución controlada de comandos, sin shell
/nmap	Simulación de escaneo por nodo
/historial	Tabla con historial ordenado y actualizable

*. Seguridad

- Comandos limitados y filtrados en servidor
- Escaping HTML con MarkupSafe
- Nmap para evitar escaneos reales peligrosos

*. Problemas y Soluciones

- Error con Matplotlib solucionado migrando a Chart.js
- Nodos Docker sin IP fija: resuelto asignando IP manual en GNS3
- NTP y DNS con errores: se configuró ntpsec y bind9 correctamente
- Plantillas embebidas dificultan mantenimiento: pendiente migración a /templates

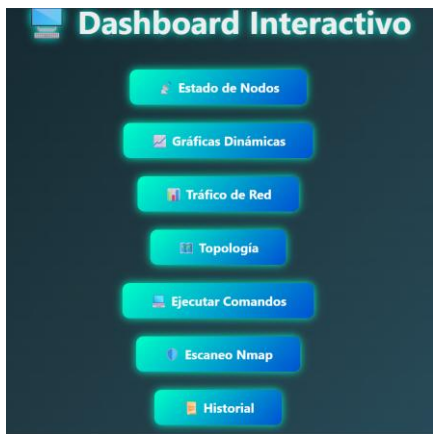
*. Conclusiones

El proyecto de monitoreo de red ha sido una experiencia técnica completa que integra redes, programación, automatización y visualización de datos. Actualmente es funcional como entorno de prueba y puede ser expandido a entornos reales con mejoras en seguridad, integración de datos reales y robustez.

*. Recomendaciones


- Integrar autenticación con Flask-Login
- Implementar consultas reales (ICMP, SNMP, NetFlow)
- Migrar HTML embebido a archivos templates
- Usar base de datos (SQLite o PostgreSQL) para historial
- Generar topología con Graphviz o Vis.js
- Agregar pruebas automatizadas con pytest

*. Evidencias



Dashboard Interactivo


- Estado de Nodos
- Gráficas Dinámicas
- Tráfico de Red
- Topología
- Ejecutar Comandos
- Escaneo Nmap
- Historial



Estado de los Nodos

DNS NTP	✓ ONLINE
Grafana	✓ ONLINE
PC Docker_1	✓ ONLINE
PC Docker_2	✓ ONLINE

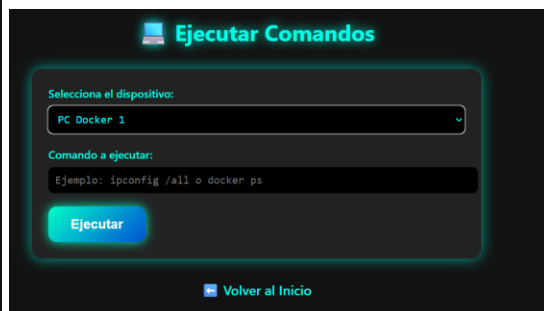
Volver al Inicio



Tráfico de Red

Nodo	Enviado (KB)	Recibido (KB)
PC Docker_1	499	566
PC Docker_2	848	901
DNS NTP	267	900
Grafana	523	767

Volver al Inicio



Ejecutar Comandos

Selecciona el dispositivo:
PC Docker 1

Comando a ejecutar:
Ejemplo: ipconfig /all o docker ps

Ejecutar

Volver al Inicio

Escaneo de Vulnerabilidades con Nmap

Selecciona el nodo:

PC Docker 1

Modo de escaneo:

Rápido (-sS)

Rápido (-sS)

Agresivo (-A)

Volver al Inicio



Ejecutar Comandos

Selecciona el dispositivo:

PC Docker 1

Comando a ejecutar:

docker ps

Ejecutar

CONTAINER ID

IMAGE

COMMAND

CREATED

STATUS

PORTS

123456789ab

nginx:latest

"nginx -g 'daemon off'"

5 minutes ago

Up 5 minutes

80/tcp

Escaneo de Vulnerabilidades con Nmap

Selecciona el nodo:

PC Docker 1

Modo de escaneo:

Rápido (-sS)

Realizar Escaneo

Realizar un report para PC Docker 1 (20-07-2025)
Realizar un report para PC Docker 2 (20-07-2025)
Realizar un report para DNS NTP (20-07-2025)
Realizar un report para Grafana (20-07-2025)

Volver al Inicio

Historial de Comandos

Hora	Nodo	Comando	Salida
2025-07-10 20:16:46	PC Docker_1	ipconfig /all	Windows IP Configuration Host Name: PC_Docker_1 IP Address: 10.0.0.11
2025-07-10 20:16:46	DNS NTP	ping -c 4	PING 10.0.0.20:58 (10.0.0.20): 56 data bytes 64 bytes from 10.0.0.20: icmp_seq=0 ttl=64 time=0.123 ms
2025-07-10 20:17:37	PC Docker_1	docker ps	CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS 123456789ab nginx:latest "nginx -g 'daemon off'" 5 minutes ago Up 5 minutes 80/tcp

Volver al Inicio

- /nodos
- /graficas
- /trafico
- /historial

- topología visual

13. Repositorio GitHub