UNIVERSIDAD MAYOR REAL Y PONTIFICA DE SAN FRANCISCO DE CHUQUISACA

FACULTAD DE TECNOLOGÍA

EXAMEN FINAL

APELLIDOS: Rodas Palacios

NOMBRE: Max Jherzon

CARRERA: Ing. En ciencias de la computación

MATERIA: Infraestructura SIS-313

CU: 111-461

SIGLA: SIS 252

DOCENTE: Ing. Lucio Marcelo

Sucre, 24 de junio de 2025

Reporte Final del Proyecto SIS313

Infraestructura de TI Tolerante a Fallos

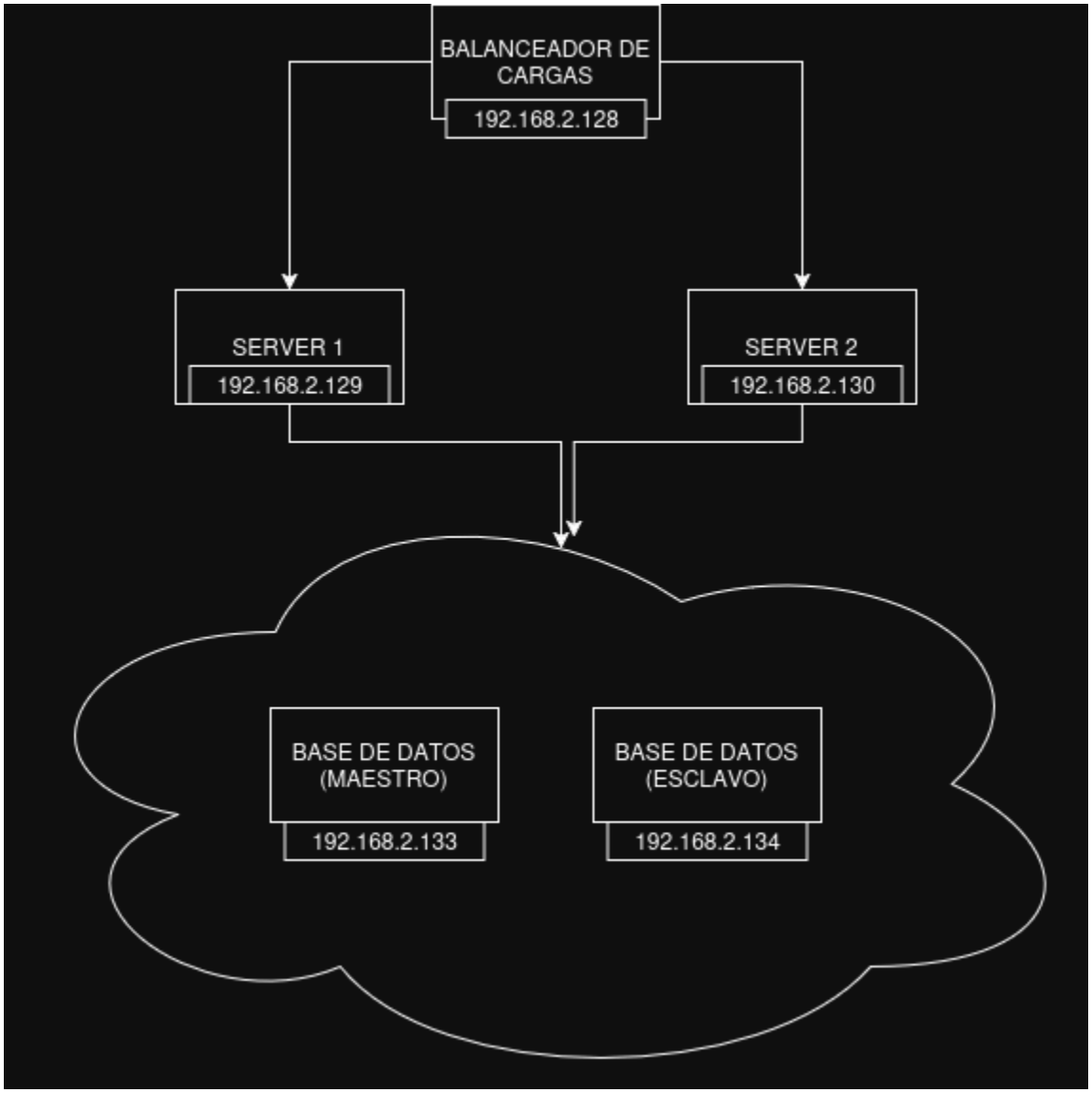
1. Descripción General del Proyecto

Este proyecto tiene como objetivo principal el diseño, implementación y demostración de una infraestructura de Tecnologías de la Información (TI) tolerante a fallos, simulando un entorno de producción empresarial. La infraestructura incluye redundancia de servicios, balanceo de carga, replicación de base de datos, seguridad, y monitoreo en tiempo real. Se utilizaron máquinas virtuales Ubuntu Server 22.04 en VirtualBox.

2. Topología de la Infraestructura

La infraestructura consta de las siguientes máquinas virtuales:

- Servidor Balanceador - 192.168.2.100
  - Sistema operativo: Ubuntu Server 22.04
  - Servicios: NGINX, Prometheus, Grafana
  - Funciones: Balanceo de carga web, monitoreo, panel de control visual
- Servidor de Aplicación 1 - 192.168.2.129
  - Sistema operativo: Ubuntu Server 22.04
  - Servicios: Node.js CRUD app, Node Exporter
- Servidor de Aplicación 2 - 192.168.2.130
  - Sistema operativo: Ubuntu Server 22.04
  - Servicios: Node.js CRUD app, Node Exporter
- Servidor de Base de Datos - 192.168.2.131
  - Sistema operativo: Ubuntu Server 22.04
  - Servicios: PostgreSQL con replicación maestro-esclavo, RAID simulado con mdadm


BALANCEADOR DE CARGAS
192.168.2.128
SERVER 1
192.168.2.129
SERVER 2
192.168.2.130
BASE DE DATOS (MAESTRO)
192.168.2.133
BASE DE DATOS (ESCLAVO)
192.168.2.134

3. Tecnologías Utilizadas

NGINX (Balanceador de carga)

Elegido por su eficiencia, facilidad de configuración y bajo consumo de recursos. Soporta múltiples métodos de balanceo como round-robin, least connections, etc., ideal para distribuir carga entre instancias Node.js.

Node.js + Express

Tecnología ligera y moderna para desarrollo backend. Permite desarrollar una API REST CRUD rápidamente y consumir una base de datos PostgreSQL sin complicaciones, ideal para simulación de servicios web.

PostgreSQL

SGBD confiable y potente que permite replicación nativa mediante configuración básica, cumpliendo el requisito de consistencia y alta disponibilidad.

mdadm + RAID

mdadm permite simular RAID 1 en entornos virtuales, asegurando disponibilidad del almacenamiento ante fallos de disco sin necesidad de hardware adicional.

Prometheus + Node Exporter

Solución de monitoreo altamente escalable y modular. Recolecta métricas de cada servidor y permite alertas. node_exporter entrega información detallada del sistema.

Grafana

Complementa a Prometheus con visualización de dashboards. Ideal para defensa del proyecto y demostración en tiempo real.

UFW + SSH Hardening

Firewall simple y efectivo para restringir el tráfico a los puertos necesarios. Las buenas prácticas de SSH como desactivar root login y cambiar de puerto ayudan a proteger los servicios del sistema.

Servicio	Herramienta	Descripción breve
Balanceador	NGINX	Distribuye carga web entre servidores de aplicación
Aplicaciones	Node.js + Express	CRUD básico conectado a PostgreSQL
Base de datos	PostgreSQL	Replicación maestro-esclavo
Monitoreo	Prometheus + Grafana	Recolección y visualización de métricas del sistema
Disco tolerante	mdadm + RAID 1	Tolerancia a fallos de disco virtual
Seguridad	UFW + SSH hardening	Reglas de firewall, puertos y usuarios seguros

4. Implementación de Servicios (Paso a Paso Detallado)

4.1 Configuración del Balanceador de Carga (NGINX)

1. Instalación:

- bash
- sudo apt update
- sudo apt install nginx -y

2. Configuración del archivo en **/etc/nginx/sites-available/balanceo**:

```
nginx
upstream backend {
        server 192.168.2.129:3000;
        server 192.168.2.130:3000;
}
server {
        listen 80;
        server_name web.sis313.usfx.bo;
        location / {
        proxy_pass http://backend;
```

```
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        }
}
```

3. Activación:

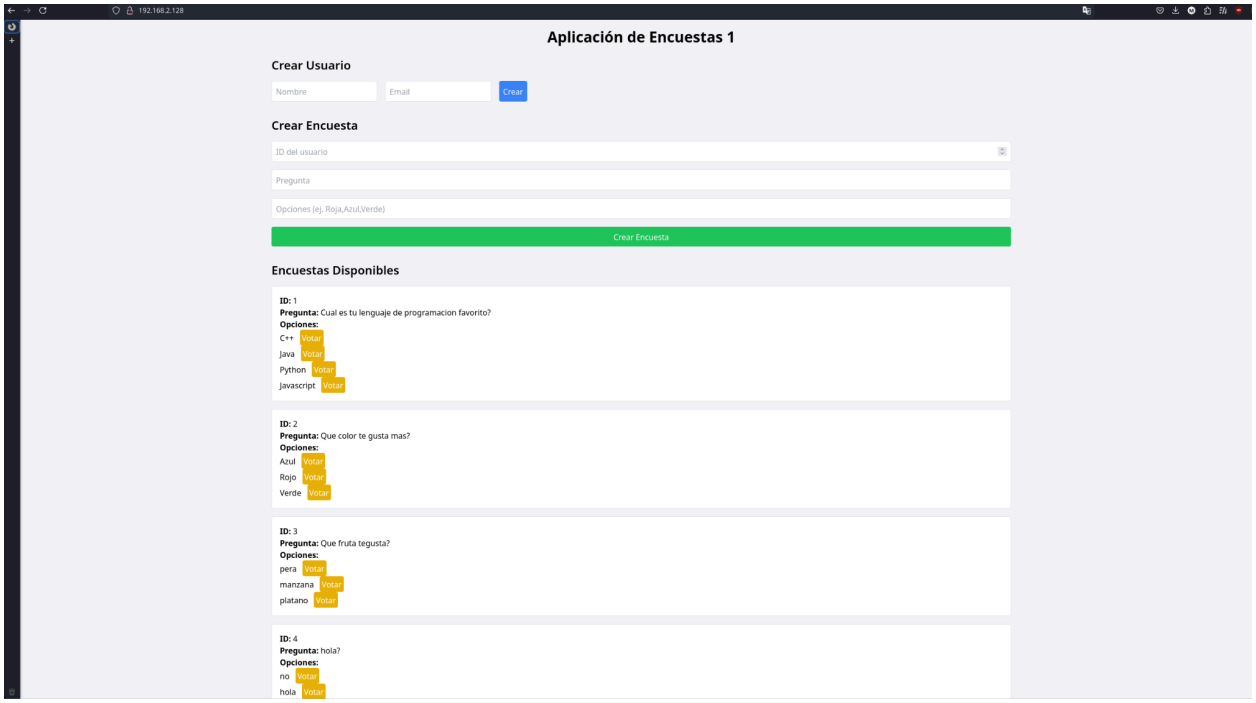
- bash
- sudo ln -s /etc/nginx/sites-available/balanceo /etc/nginx/sites-enabled/
- sudo systemctl reload nginx

4.2 Aplicaciones Web (Node.js)

1. Instalación de Node.js:

- bash
- curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -
- sudo apt install nodejs -y

2. Estructura de la app CRUD:



3. Lanzamiento:

- bash
- node index.js

4.3 PostgreSQL con Replicación Maestro-Esclavo

1. Configurar `postgresql.conf` en el maestro:

{

conf

wal_level = replica

archive_mode = on

max_wal_senders = 3

}

2. Configurar `pg_hba.conf`:

- conf
- host replication replicador 192.168.2.130/32 md5

3. Crear usuario y permisos:
   - sql
   - CREATE USER replicador REPLICATION LOGIN ENCRYPTED PASSWORD 'replicapass';
4. En esclavo:
   - bash
   - pg_basebackup -h 192.168.2.131 -D /var/lib/postgresql/15/main -U replicador -P -R

4.4 RAID 1 con mdadm

1. Agregar dos discos virtuales desde VirtualBox.
2. Instalar mdadm:
   - bash
   - sudo apt install mdadm -y

3. Crear RAID:
   - bash
   - sudo mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sdb /dev/sdc
   - sudo mkfs.ext4 /dev/md0
   - sudo mount /dev/md0 /mnt/raid
4. Simular falla:
   - bash
   - sudo mdadm --fail /dev/md0 /dev/sdb
   - sudo mdadm --remove /dev/md0 /dev/sdb
   - sudo mdadm --add /dev/md0 /dev/sdb

4.5 Monitoreo con Prometheus y Grafana

1. Instalar Prometheus:
   - bash
   - wget https://github.com/prometheus/prometheus/releases/download/v2.52.0/prometheus-2.52.0.linux-amd64.tar.gz

2. Configurar `/etc/prometheus/prometheus.yml`:

{

yaml

scrape_configs:

  - job_name: 'servidores-sis313'

        static_configs:

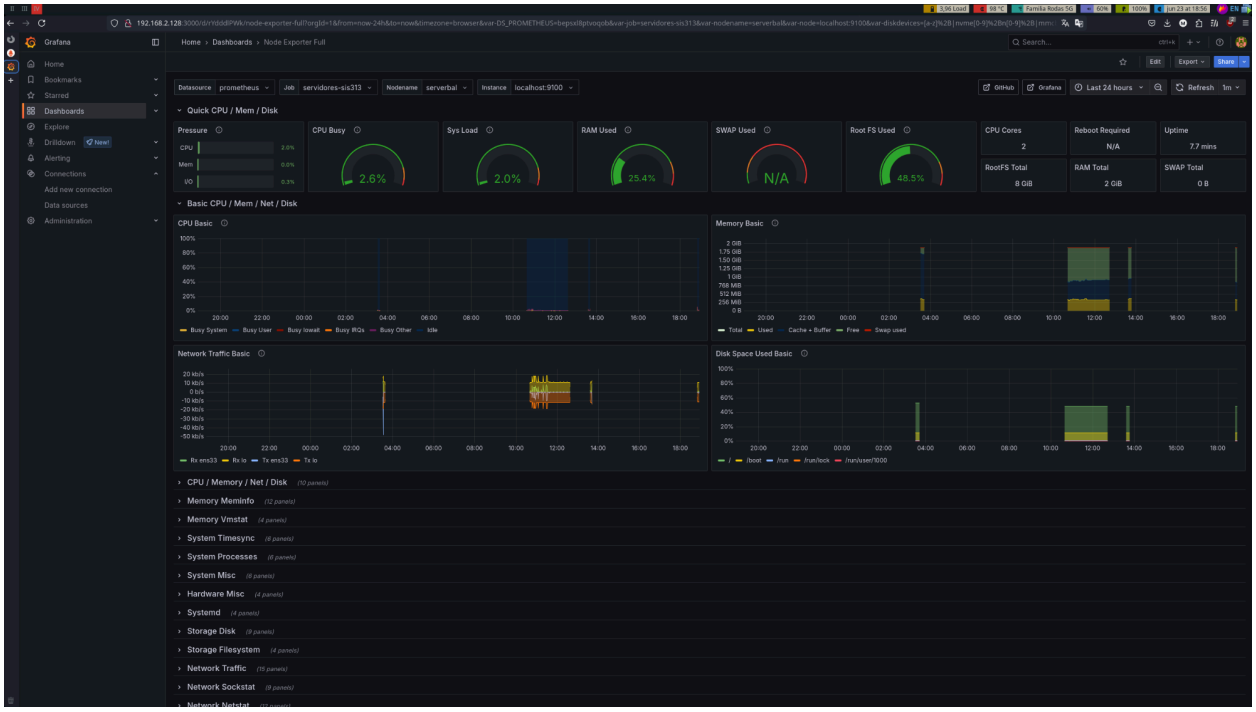
        - targets: ['localhost:9100']

}

3. Instalar Node Exporter:
   - bash
   - wget https://github.com/prometheus/node_exporter/releases/download/v1.8.1/node_exporter-1.8.1.linux-amd64.tar.gz

4. Instalar Grafana:
   - **bash**
   - **sudo apt install -y grafana**
   - **sudo systemctl start grafana-server**

5. dashboard en Grafana con ID 1860.



5. Seguridad y Hardening

- SSH con puerto personalizado (**Port 2222**).
- Deshabilitación de **PermitRootLogin.**
- Autenticación mediante clave pública.
- Reglas `UFW` para permitir solo puertos: 2222, 80, 3000, 5432, 9090, 9100, 3000.

6. Evidencia de Tolerancia a Fallos

- Se apagó un servidor de aplicación y el balanceador mantuvo el servicio activo.
- Se simuló falla de disco en RAID y el sistema siguió operativo.
- Replicación de PostgreSQL demostrada con inserciones desde maestro y consultas desde esclavo.

7. Conclusiones

El proyecto integró diversas áreas de la carrera: redes, sistemas operativos, seguridad, servicios web, virtualización y monitoreo. Se construyó una infraestructura realista, escalable y resistente a fallos.

8. Repositorio y Video

Repositorio GitHub: [https://github.com/M4x241/Infraestructura-Tolerante-Fallos.git)

Video_Demostrativo:

[https://drive.google.com/file/d/1I_C5FUIYp339ryubyQ6oDaSkElXZIp-m/view?usp=sharing]