LABORATORIO

Proyecto Final

preparado por Prof. Antonio Russoniello

Monitoreo de Redes y Telemetría por Protocolo MQTT usando Bot de Telegram

1 Descripción General

Este proyecto tiene como objetivo desarrollar un sistema que permita a un usuario, a través de un bot de Telegram, ejecutar comandos desde un servidor para el monitoreo de la latencia y los saltos a un nodo objetivo. Los resultados de estas pruebas, además de ser mostrados por el bot desde la inteerface de Telegram también serán publicados en un broker MQTT bajo un tópico específico. Adicionalmente, el sistema monitoreará de forma recurrente, un nodo especifico y enviará una alerta al bot de Telegram si el mismo se encuentra en estado inalcanzable.

2 Objetivos Específicos

1. Desarrollar un Bot de Telegram que reciba instrucciones del usuario para ejecutar comandos de red desde un servidor a un nodo objetivo.
2. Implementar librerías de Python para la ejecución de los comandos de red hacia un objetivo especificado por el usuario y analizar la salida para extraer los datos que serán presentados como respuesta (por ejejmplo latencia promedio y saltos).
3. Configurar un cliente MQTT en Python para conectarse a un broker MQTT y publicar los resultados de cada comando en el tópico **"mensaje_grupo"**.
4. Graficar en MQTT Explorer un gráfico de serie de tiempos con los resultados de la latencia y los resultados de la cantidad de saltos.
5. Diseñar el sistema de forma modular para facilitar su comprensión.

3 Arquitectura del Sistema

El sistema constará de los siguientes componentes principales:

1. Bot de Telegram: Interactúa con el usuario para recibir comandos y enviar respuestas y alertas.
2. Módulo de Captura de Comandos: Ejecuta el comando y parsea la salida para obtener el resultado (por ejemplo comando ping y como resultado la latencia promedio).
3. Cliente MQTT: Se conecta al broker MQTT, publica los resultados obtenidos al broker.
4. Broker MQTT: Para la recepción de los distintos mensajes (tópicos) generados por el cliente.
5. Módulo de monitoreo: Presentación de los resultados obtenidos utilizando el Explorador MQTT.

4 Tecnologías a Utilizar

1. Python: Lenguaje de programación principal para el script que se ejecutará en el servidor para interactuar con el bot de Telegram, para ejecutar los comandos de red, para implementar el cliente MQTT y para implementar la lógica de alertas.
2. Librería Python para Telegram: Necesaria para interactuar con la API de Telegram, por ejemplo: python-telegram-bot o cualquier que decida.
3. Librería paho-mqtt: Para la comunicación con el broker MQTT.
4. Broker MQTT: Se puede utilizar un broker público como broker.hivemq.com o test.mosquitto.org, esto para efectos de la realización de pruebas, sin embargo, el desarrollo final debe ejecutarse en un broker local.
5. Sistema Operativo: Linux o Windows.

En la Fig. 1 se presenta una arquitectura referencial con todos los elementos de red y los componentes necesarios.

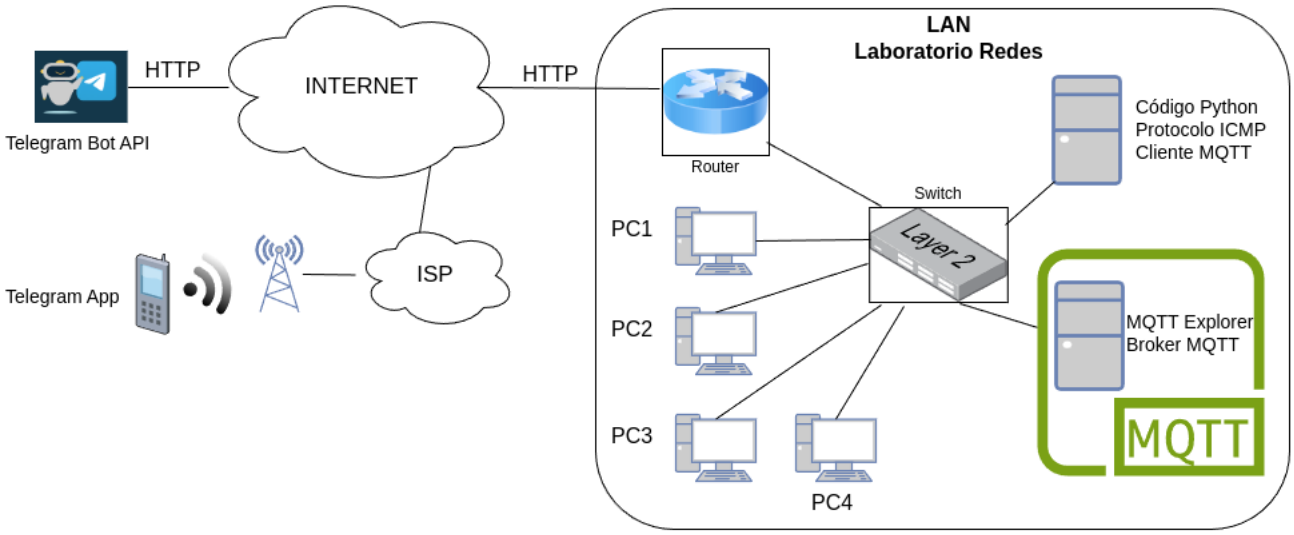

Figura 1: Arquitectura Referencial

5 Plan de Desarrollo

5.1 Desarrollo del Bot de Telegram

1. Crear un bot y obtener su token de acceso.
2. Implementar el comando [destino] para que el bot reciba la dirección IP o nombre de host del objetivo.

3. Crear un script en Python que tome el destino como entrada y ejecute por lo menos 4 solicitudes via ping utilizando la librería subprocess. Se recomienda implementar un retardo de 5 segundos antes de volver a reiniciar el ciclo de captura. También debe mostrarse la cantidad de saltos para alcanzar al nodo destino.

4. Analizar la salida del comando ping para extraer el tiempo de respuesta (latencia) promedio (de las 4 solicitudes ping). Esta información, junto a la información de saltos debe enviarse al bot y al explorador MQTT para efectos de monitoreo.

5. Adicionalmente debe implementarse en el bot una instrucción que inicie de forma recurrente consultas desde el servidor a un nodo objetivo. Unicamente en caso de no existir conexión al host objetivo, debe reportarse al bot mediante un mensaje de alerta. La verificación debe ejecutarse de forma recurrente y finalizará solo cuando, mediante instrucción explicita desde el bot, sea detenida. Note que este proceso es independiente del proceso de monitoreo de los pasos anteriores.

6. Implemente una instrucción de parada, que permita, desde el bot, detener las capturas.

5.2 Integración con Broker MQTT:

1. Configurar un cliente MQTT utilizando la librería paho-mqtt.

2. Implementar la conexión al broker MQTT.

3. Crear una función para publicar la latencia y los saltos obtenidos en los tópicos **“mensaje_grupo”**.

5.3 Implementación del Monitoreo:

1. Crear una función que reciba los mensajes publicados en el tópico “mensaje_grupo”.

2. Extraer el valor de promedio de la latencia y la cantidad de saltos, mostrar esta información en el bot.

3. Implementar una condición para verificar si el host objetivo se encuentra desconectado o inalcanzable, en este último caso, enviar un mensaje de alerta al bot de Telegram, indicando el destino y un mensaje de alerta.

4. Los valores de latencia y de los saltos deberán ser graficados en serie de tiempo en el explorador MQTT.

5.4 Pruebas y Ajustes

1. Realizar pruebas exhaustivas del sistema, verificando la correcta funcionalidad del bot, la captura de los resultados, la publicación en MQTT y el sistema monitoreo gráfico en el explorador MQTT.

2. Ajustar los parámetros y la lógica según sea necesario.

3. Para sus pruebas considere desplegar una máquina virtual para el host objetivo o ejecute las pruebas hacia la computadora que le indique el instructor del laboratorio.

6 Seguridad y Captura de Errores

1. Para la configuración del broker MQTT, tener en cuenta las implicaciones de seguridad y considerar el uso de autenticación por usuario y clave.

2. Implementar captura de errores para situaciones como fallas en la conexión de red, errores al ejecutar los comando de red o problemas con el broker MQTT.

3. Considerar el impacto de ejecutar comandos los comandos de red repetidamente en el servidor. Para evitar condiciones de bloqueo, se recomienda que el ciclo de ejecución de los comandos de red sea superior a los 5 segundos.

7 Presentación de Resultados y Evaluación

El día de la entrega final, debe mostrar sus resultados para los siguientes escenarios:

1. Inicio del servicio desde el bot: Desde el bot debe poder iniciarse la captura de los valores de latencia y de los saltos. **(20%)**

2. Monitoreo por Telegram: Se verificará que se reciban los valores de promedio de latencia, los saltos y las alertas ante host inalcanzable. **(30%)**

3. Monitoreo por serie de tiempo: Se deberán mostrar los valores de latencia y saltos cada uno en un gráfico de serie de tiempos en el explorador MQTT. **(30%)**

4. Parada del servicio: Desde le bot de Telegram deberá poder detenerse tanto el servicio de monitoreo y como el de alerta. **(20%)**