Internet de las Cosas 2024

Informe del trabajo final.

Integrantes del grupo

Olmos Joaquín, 18707/4
Panigo Ailén, 18740/5
Vaccaro Paula, 17139/8

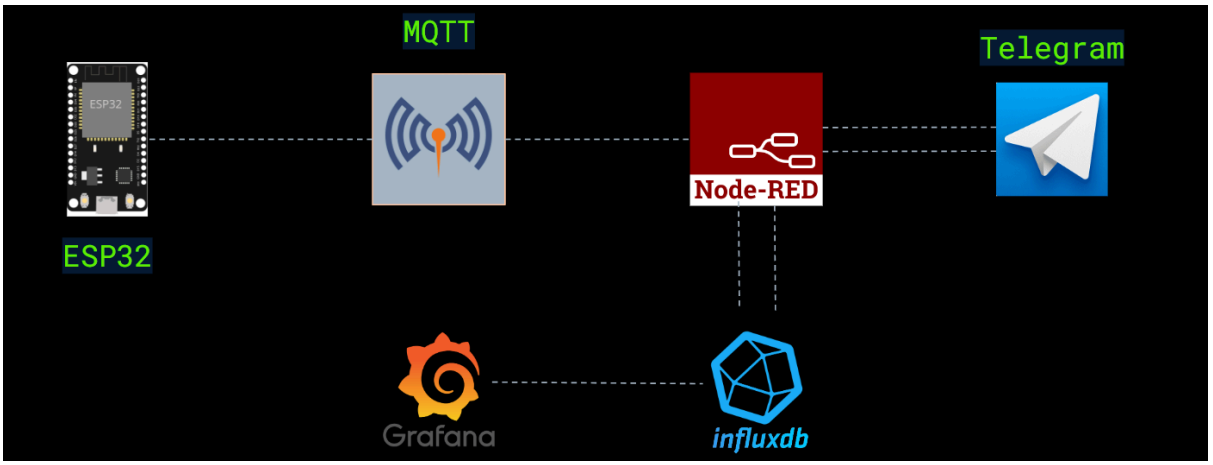
Informe

Sensado de temperatura en un invernadero, con la posibilidad de usar un bot interactivo en Telegram y visualizar los datos históricos en un tablero de Grafana.

Introducción

Modificamos un poco lo planteado en la propuesta enviada. Se realizó el desarrollo de un nodo sensor de temperatura en un invernadero que abre y cierra ventanas dadas ciertas temperaturas. Las temperaturas generadas se almacenan en una base de datos que permite su análisis y muestra en representaciones gráficas y, al mismo tiempo, también se generó un bot con el que se permiten tener actualizaciones de la temperatura y el estado de las ventanas, así como también abrir y cerrar estas últimas a través de comandos.

Diagrama de solución



ESP32

Utilizamos el ESP32 para simular la detección de temperaturas, esto lo hace generando un número aleatorio entre 10 y 40 y publica el mensaje en el broker MQTT. Cada mensaje es un registro de temperatura. La temperatura es enviada cada 5 segundos por cuestiones de testeo, pero lo ideal sería que se realice cada media hora aproximadamente.

MQTT

Para este proyecto, utilizamos Mosquitto como el broker MQTT. Inicialmente intentamos levantar Mosquitto de manera local utilizando Docker, pero debido a problemas de conectividad en nuestra red local, optamos por utilizar el broker público. Teniendo en cuenta que el broker es público, tuvimos que adaptar el tópico para que sea lo suficientemente "complejo" y no recibir mensajes ajenos a la temperatura enviada por el ESP. Por esto, el tópico elegido fue esp/supercomplex/system/temperature.

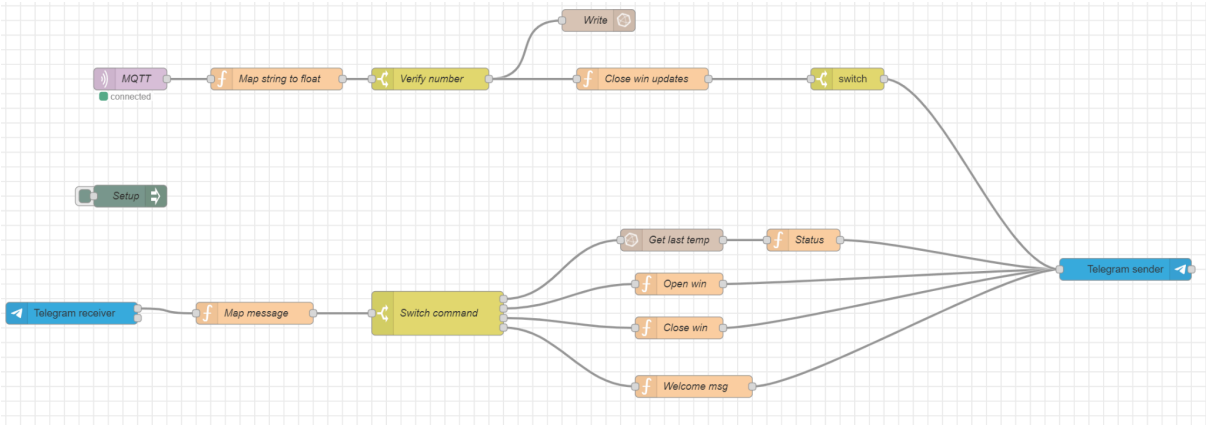
A partir de esto otras herramientas se pueden suscribir al tópico para recibir los datos. En nuestro caso, esta sería Node-RED.

Node-RED

Utilizamos Node-RED para establecer el flujo del sistema e intermediario para todas las herramientas que utilizamos. El nodo Setup se encarga de inicializar las variables utilizadas durante todo el flujo, las cuales son: un array de ids de chats a los cuales el bot va a enviar las actualizaciones, un bool motorActivo que indica el estado del motor y otro bool ventanasAbiertas que indica si las ventanas están abiertas o no.

Por otro lado, se reciben los datos de temperatura desde el servidor MQTT y los almacena en la base de datos InfluxDB. Con la misma temperatura recibida, se verifica el estado de las ventanas para saber si es necesario abrir o cerrarlas, siempre teniendo en cuenta si el motor está activo o no, para no sobrecargarlo. Si se cambió efectivamente el estado de las ventanas, se notifica por Telegram.

Al mismo tiempo, Node-RED se integra con Telegram para permitir la interacción del usuario con un bot creado especialmente con ese objetivo. Esta interacción implica tres comandos principales, y si se recibe cualquier otro mensaje por parte del usuario, el bot envía un mensaje personalizado de bienvenida.

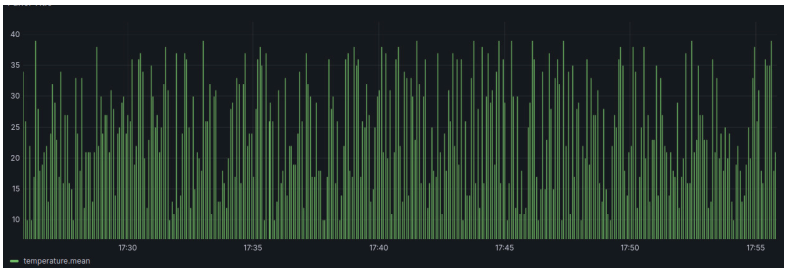


InfluxDB

InfluxDB nos sirve para guardar los datos de la temperatura. Lo hace dejando una marca de tiempo para cada dato. Con esta herramienta, almacenar las temperaturas de esta forma nos permite hacer un seguimiento histórico que permite su análisis.

Grafana

Grafana se integra con InfluxDB para poder capturar todos los datos que se almacenan en la base de datos y, con ellos, generar las representaciones gráficas de los cambios de temperatura que se generaron.



Grafana también nos permite generar alertas para cuando, por ejemplo, las temperaturas superan cierta cantidad de grados.

Telegram

Usamos telegram para definir el bot que envía las actualizaciones. Como ya mencionamos anteriormente, Node-RED envía al bot la información de temperatura que se registra desde MQTT y las envía a los usuarios que estén habilitados para recibir estas actualizaciones.

Al mismo tiempo, a través de Telegram los usuarios pueden enviar órdenes al sistema. Estos son:

- **/estado**: le permite al usuario obtener el estado actual. Esto implica la temperatura actual y el estado de las ventanas. Para obtener la temperatura actual, el flujo se encarga de obtener de InfluxDB la última temperatura guardada.
- **/abrir**: le permite al usuario abrir las ventanas del invernadero. Esto se realiza sólo si el motor no se encuentra activo, sea por una solicitud del usuario o por un cambio automático por la temperatura obtenida, o si las ventanas están cerradas.
- **/cerrar**: le permite al usuario cerrar las ventanas del invernadero. Esto se realiza sólo si el motor no se encuentra activo o si las ventanas están abiertas.
- De enviar otro mensaje al bot, muestra un mensaje de bienvenida con las opciones disponibles para poder interactuar.

Desafíos

Definimos un par de desafíos que nos hicieron demorar el desarrollo del sistema:

- Ya fue descripto al mencionar la implementación de MQTT, pero al intentar usar el broker MQTT de forma local nos encontramos con problemas de conexión por parte del ESP; este último no encontraba el broker MQTT levantado con Docker. Esto se daba por una variedad de problemas, entre ellos los proxys configurados en la red local, que el wifi no esté definido sobre la misma IP que el ethernet, etc. Por ese motivo optamos por buscar un servidor alojado públicamente para poder continuar. No tuvimos problema luego de hacerlo de esa manera. Igualmente generamos un topic lo suficientemente complejo como para evitar las interferencias de otros usuarios que también estén usando el servidor.
- Los permisos de Telegram estaban definidos de forma tal que sólo se tomaban mensajes en formato de comando (con la barra / al principio), y no cualquier tipo de mensaje. Para esto tuvimos que cambiar la configuración de privacidad desde el BotFather para que permita todo tipo de mensajes.

Desarrollos Futuros

Se podrían agregar módulos para sensar las diferentes variables pertinentes en un invernadero, como temperatura, luz y humedad del suelo y lograr así automatizar el adecuado cuidado de las plantas e implementar un sistema de alertar cuando se superan ciertos valores que podrían generar daños en la planta o alertar de funcionamientos anormales en el sistema.

Además, se podría generar un modelo de ML alimentado con los datos recolectados que produzca alertas preventivas en caso de hongos o plagas.

Utilizar un segundo ESP-32 para manejar la interacción con el bot de telegram desacoplando de Node-RED.

Aprendizajes Adicionales

A raíz de llevar a cabo este proyecto logramos comprender en mayor profundidad conceptos que nos benefician en nuestro desarrollo profesional.

Ataques de cadena de suministro:

Los ataques de cadena de suministro se producen cuando un tercero inyecta código malicioso en las librerías que se importan en un proyecto. Aunque un sistema pueda parecer libre de código malicioso, este puede ser introducido a través de librerías de confianza.

Estos ataques suelen aprovecharse de la confianza que los desarrolladores depositan en las librerías y dependencias externas. Un ejemplo reciente es el caso de la librería XZ Utils, utilizada en muchos sistemas Linux. Esta librería sufrió una inyección de código malicioso, afectando a numerosos sistemas de versiones rolling release.

Nuestra experiencia nos enseñó la importancia de revisar rigurosamente las librerías antes de integrarlas en nuestros proyectos. En nuestro caso, por la presión de cumplir con los plazos de entrega, no verificamos si las librerías habían sido comprometidas, lo que nos hizo vulnerables a este tipo de ataques.

Estos incidentes subrayan la necesidad de implementar medidas de seguridad robustas y procedimientos de auditoría para las librerías y dependencias que utilizamos. Asegurarse de que las librerías no han sufrido modificaciones no autorizadas es crucial para mantener la integridad y seguridad de nuestros sistemas.

IoT e Ingenieria de datos:

Hoy en día, se dice que los datos son el "oro" de nuestra época. Con la globalización y el avance de la tecnología, todos nuestros dispositivos generan datos constantemente. Estos datos son recolectados a partir de diversas fuentes, como teléfonos, computadoras, smartwatches, televisores y navegadores web.

El Internet de las Cosas (IoT) ha jugado un papel crucial en esta transformación. IoT se refiere a la interconexión de dispositivos y objetos a través de Internet, permitiendo que recojan y compartan datos. Estos dispositivos pueden ser desde simples sensores hasta electrodomésticos inteligentes y vehículos autónomos.

La ingeniería de datos es la disciplina encargada de diseñar, construir y mantener los sistemas que permiten gestionar y analizar estos grandes volúmenes de datos. Los ingenieros de datos crean infraestructuras que facilitan el almacenamiento, procesamiento y análisis eficiente de los datos generados por dispositivos IoT y otras fuentes.

El análisis de estos datos permite a las empresas obtener información valiosa sobre el comportamiento de los usuarios, optimizar operaciones y crear nuevos modelos de negocio. Por ejemplo, en el ámbito de la salud, los datos recolectados por dispositivos wearables

pueden ayudar a monitorizar la salud de los pacientes en tiempo real y a prevenir enfermedades mediante análisis predictivos .

Conclusiones

Realizar este proyecto de IoT nos permitió aplicar de manera práctica y efectiva los conocimientos adquiridos en la materia. En retrospectiva, nos hubiera gustado desarrollar algo más complejo, idea que se vio afectada por la carga académica y profesional de todos los integrantes del grupo. A pesar de la limitación temporal, estamos satisfechos con los resultados obtenidos y valoramos enormemente la oportunidad de integrar diversas tecnologías.

El proyecto reforzó nuestros conocimientos técnicos y nos brindó una comprensión más profunda sobre la gestión del tiempo y el trabajo en equipo. Consideramos muy interesante la posibilidad de, en el futuro, expandir este proyecto para incluir sensores reales y evaluar su rendimiento en un entorno práctico.

Referencias

- OWASP. (2023). Inadequate supply chain security. OWASP. https://owasp.org/www-project-mobile-top-10/2023-risks/m2-inadequate-supply-chain-security
- Synopsys. (n.d.). Xz utils backdoor: A supply chain attack. Synopsys. https://www.synopsys.com/blogs/software-security/xz-utils-backdoor-supply-chain-attack.html
- Mosquitto. (n.d.). Test Mosquitto. Mosquitto. https://test.mosquitto.org/
- Node-RED. (n.d.). User guide. Node-RED. https://nodered.org/docs/user-guide/
- InfluxData. (2018). IoT made easy with Node-RED and InfluxDB. InfluxData. https://www.influxdata.com/blog/iot-easy-node-red-influxdb/