UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Proyecto Final

2021

Monitoreo de alarmas para Datacenter

Tesina

Casas Daiana, Pregelj José, Riveros Ruben, Vieiro Alexis

Índice

1. Objeto del Documento

Este documento tiene por objeto la explicación detallada y con el máximo rigor técnico los conceptos teóricos de aplicación en el proyecto, con referencias bibliográficas que cubran los contenidos teóricos clásicos, la cual debe ser actualizada y de reconocido prestigio.

2. Abstract

La presente solución esta destinado a Datacenters. Entendiendo que es un ambiente físico dedicado a alojar equipos y equipamiento físicos; donde se necesita un sistema de monitoreo para prever eventos o atenderlos y generar acciones correctivas o un tratamiento adecuado. Encontramos la oportunidad de ofrecer un sistema de monitoreo contemplando y reutilizando distintos tipos de sensores que pueda poseer dicha instalación dentro un mismo producto y servicio sin tener que adquirir sensores específicos de alto precio o bajo stock dentro del país.

3. Introducción Teórica del tema a tratar y Marco Teórico

A continuación se abarcará distintos conceptos que permiten comprender la solución como el problema.

3.1. Datacenters

Definimos *Datacenter* a aquel ambiente físico dedicado a alojar equipos físicos, ya sean servidores, conexiones y otros recursos necesarios para mantener una red o un sistema de computadoras de información

Este ambiente físico son salas donde se disponen todos los equipos acomodados en estructuras de forma organizada, llamada racks (Fig. 1).

Figura 1: Instalaciones de datacenter

Algunos *Datacenters* pueden tener una sala de energía que mantiene energizado todo el equipamiento en todo momento. También pueden tener una sala de control (Fig. 2) de los equipos, generalmente son los administradores de los mismos, llamada *NOC* (Network Operations Center).

3.2. Monitoreo

Es importante conocer el estado de nuestros equipos, recursos y servicios para prever eventos o atenderlos y generar acciones correctivas o un tratamiento adecuado. Esta visualización constante de estos estados, mensajes o cualquier otro evento en tiempo real se llama monitorización.

Figura 2: Monitorización

Existen desarrollos de software que nos permiten realizar estas tareas, que se denominan *NMS* "Network Management Software". Este es el encargado de

recopilar la información, procesarla y hace la disponible para que el usuario interactúe con aquello que esta monitorizando.

Algunos ejemplos de *NMS* actuales son Zabbix, Nagios, OpenNMS, PandoraFMS, The Dude, PRTG Network Monitor.

3.3. SNMP

SNMP (Simple Network Management Protocol) es un protocolo estandarizado para la capa de aplicación (Fig. 3). Tiene una arquitectura cliente - servidor donde los agentes/clientes son los equipos y el servidor es el *NMS*.

Dentro de este intercambio de información hay distintas formas de realizarlo, las cuales se detallan a continuación.

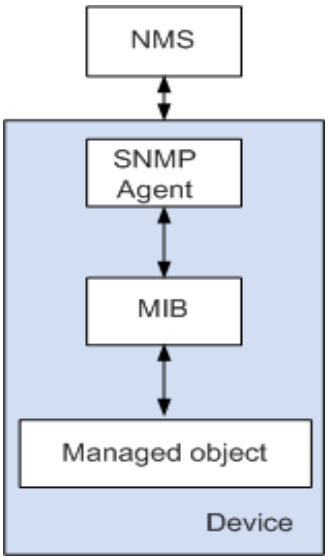

Figura 3: Protocolo SNMP

3.3.1. GET

GET es una solicitud enviada por el servidor *NMS* al equipo. Luego de recibir el *GET* el agente *SNMP* ejecuta la instrucción especificada en la solicitud y correspondiente en el *MIB* para luego responder con el resultado al *NMS* (Fig. 4). Las operaciones que pude realizarse en *SNMP GET* incluye Get, GetNext y GetBulk.

- Get: esta operación permite al *NMS* obtener una o más variables desde el *SNMP* agente.
- GetNext: esta operación permite al *NMS* obtener una o mas variables subsecuentes desde el *SNMP* agente.

- GetBulk: esta operación es igual que ejecutar consecutivos GetNext. Se puede setear la cantidad de operaciones que implica en la operación GetBulk.

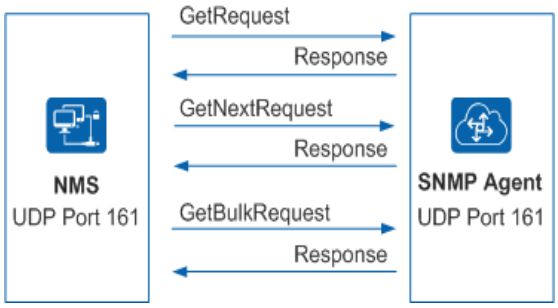

Figura 4: Proceso GET

En la figura 5 se observa el formato de los paquetes de la operación *GET SNMP*

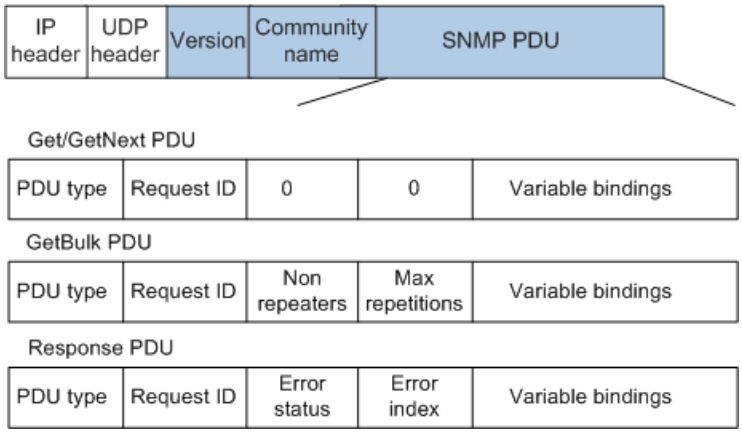

Figura 5: Contenido del paquete GET

A continuación se detallan los campos del paquete

- Versión: especifica la versión de *SNMP* a utilizar. El valor 0 es para la versión 1 y 1 para la versión 2c.
- Community name: especifica la autenticación entre el agente *SNMP* y el *NMS*.Este valor es de tipo texto.
- Request ID: especifica un único ID de cada solicitud realizada. Se utiliza para la correspondencia solicitud - respuesta.
- Non repeaters/Max repetitions: La operación GetBulk es igual a realizar operaciones consecutivas GetNext. Ambos campos permiten setear el numero de operaciones GetNext.
- Error status: especifica el estado de un error que ocurre en el procesamiento de la solicitud.

- Error index: especifica el índice de un error. Si ocurre una excepción, se especifica información de la causa de la excepción.
- Variable Bindings: especifica la lista de nombres de variables y valores correspondientes a la operación.

3.3.2. SET

El sistema *NMS* puede enviar solicitudes al agente de *SNMP* para configurar se utiliza para modificaciones de valores de información para el control del equipo (Fig. 6).

El agente *SNMP* ejecuta la instrucción correspondiente y especificada en el *MIB*, para luego retornar el resultado al *NMS*.

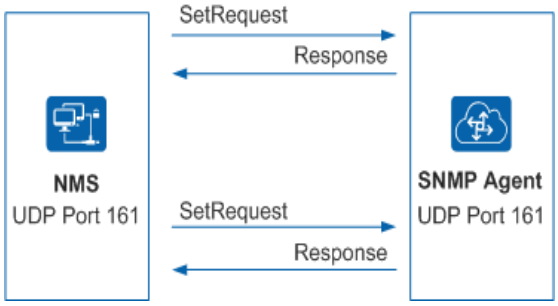

Figura 6: Proceso SET

En la figura 7 se observa el formato de los paquetes de la operación *SET SNMP*

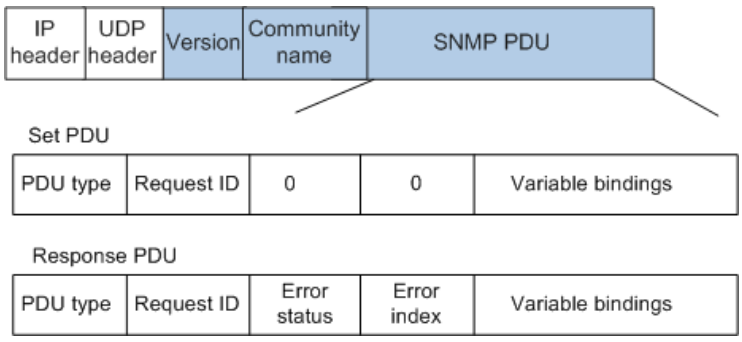

Figura 7: Contenido del paquete SET

A continuación se detalla el paquete en cuestión

- Version: especifica la versión de *SNMP* a utilizar. El valor 0 es para la versión 1 y 1 para la versión 2c.
- Community name: especifica la autenticación entre el agente *SNMP* y el *NMS*. Este valor es de tipo texto.

- Request ID: especifica un único ID de cada solicitud realizada. Se utiliza para la correspondencia solicitud - respuesta.
- Error status: especifica el estado de un error que ocurre en el procesamiento de la solicitud.
- Error index: especifica el índice de un error. Si ocurre una excepción, se especifica información de la causa de la excepción.
- Variable Bindings: especifica la lista de nombres de variables y valores correspondientes a la operación.

3.3.3. TRAP

Los *TRAP* son mensajes de notificación que envía por el agente *SNMP* hacia el sistema *NMS* para informar alarmas o eventos generados por el equipo. Hay dos tipos de esta operación: *TRAP* e *INFORM* (Fig. 8). La diferencia entre ellos es la existencia de respuesta del sistema *NMS*.

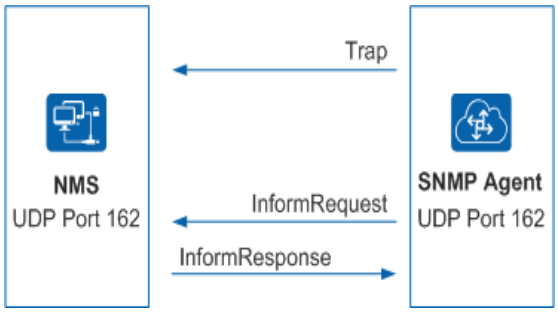

Figura 8: Proceso TRAP

En la figura 9 se observa el formato de los paquetes de la operación *GET SNMP*

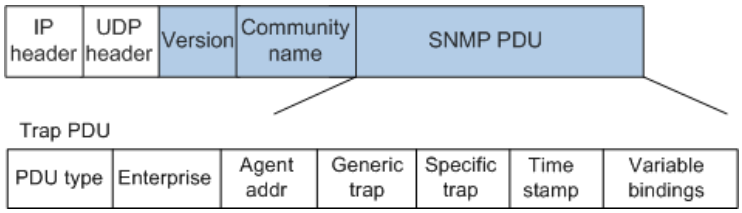

Figura 9: Contenido del paquete TRAP

A continuación, se detalla el paquete utilizado para esta operación

- Version: especifica la versión de *SNMP* a utilizar. El valor 0 es para la versión 1 y 1 para la versión 2c.
- Community name: especifica la autenticación entre el agente *SNMP* y el *NMS*.Este valor es de tipo texto.
- Enterprise: especifica el tipo de *TRAP* que genera e equipo.

- Generic trap: especifica un tipo de *TRAP* común, incluyendo coldStart, warmStart, linkDown, linkUp, authenticationFailure, egpNeighborLoss y enterpriseSpecific.
- Specific trap: especifica el *TRAP* privado con su correspondiente información.
- Time stamp: especifica el tiempo transcurrido entre el tiempo cuando la entidad de red es reiniciada y el tiempo cuando el mensaje de *TRAP* es generado.
- Variable Bindings: especifica la lista de nombres de variables y valores correspondientes a la operación.

3.4. Puertos utilizados

El protocolo *SNMP* utiliza, comúnmente, el puerto de tipo UDP. El mismo puede ser por default:

- Puerto 161: este puerto es utilizado por el *NMS* para las operaciones de solicitud Get, GetNext, GetBulk, y Set. Sobre este mismo puerto responde el agente SNMP a esas operaciones.
- Puerto 162: este puerto es utilizado por el agente *SNMP* para enviar los *TRAPs* o mensajes con información hacia el *NMS*. Este puerto es configurable, este debe ser configurado en ambos extremos de la comunicación.

3.5. MIB

Management Information Base , contiene las variables que maneja y permite el equipo. El archivo *MIB* define los atributos manejados por el equipo, incluyendo el nombre,estado y demás. Esto permite tener una interfaz entre la cola de mensajes del *NMS* y las variables del equipo.

Este archivo contiene una estructura que mantiene el almacenamiento de la información. Se conforma por un árbol con nodos que determinan un único objeto e identificable .

4. Desarrollo de la solución a implementar

El proyecto desarrollado es un servicio de monitoreo de distintos tipos de dispositivos conectados a una red incluyendo un equipo que me permite adquirir distintos tipos de mediciones del mismo ambiente.

El objetivo principal será explotar los distintos tipos de datos recolectados de forma visual e intuitiva para los usuarios finales. Se incluirá la emisión de alarmas para cualquier tipo de evento que suceda. Se utilizarán tecnologías actuales de punta y atendiendo las necesidades de los clientes que hoy posee, orientándonos a la experiencia del usuario para mejorar la calidad de sus actividades.

Se utilizará principalmente el protocolo de red *SNMP* v2, el cual recolecta de estos equipos, datos como el uso de CPU, RAM, porcentaje de disco utilizado, tráfico actual, temperatura, nivel de tensión de alimentación, etc. Los principales dispositivos a considerar son centrales telefónicas, switches, routers, access points, impresoras, servidores, computadoras, equipo de radioenlace o cualquier otro equipo/dispositivo que posea este protocolo.

Este servicio queda abierto a distintos equipos que pueda tener el cliente, siempre y cuando los mismos soporten el protocolo *SNMP* o al menos respondan a paquetes *PING*. Agregando la posibilidad de integrar los sensores que posean, con las mismas características de comunicación.

Este servicio va a contar con un servicio web y una aplicación de celular para el acceso directo e inmediato a la información de la red en cuestión.

El sistema de monitoreo presentado va a ofrecer una combinación de arquitecturas de infraestructura,conformado por:

- un sistema *NMS* en la nube
- un sistema instalado en la casa del cliente

El primero para poder acceder al sistema sin necesidad de mantener una conexión hacia la red del cliente , evitar enfrentarse con restricciones de la red del cliente o el hardware necesario para ponerlo en funcionamiento, entre otros. El segundo para la comunicación y utilización de distintos equipos y tipos de sensores con el sistema central en la nube, es decir el primero.

4.1. Alcance

Sistema de monitoreo

Este servicio queda abierto a distintos equipos que pueda tener el cliente, siempre y cuando los mismos soporten el protocolo *SNMP* o al menos respondan a paquetes *PING*. Agregando la posibilidad de integrar los sensores que posean, con las mismas características de comunicación.

Concentrador de sensores

Dispositivo electrónico de diseño propio que se encargará de suministrar la información al servidor central en la nube. Para el diseño de la solución se define las siguientes especificaciones:

- poseer 1 entradas de sensor de temperatura
- poseer 4 entrada de accion de puerta
- conector RJ45 para conexión y transmisión de información.

4.2. Etapas de proyecto

Para realizarlo se plantearon 3 hitos importantes, los cuales se dividen en los siguientes puntos:

- Integración con la nube donde se instalará el sistema *NMS*
- Concentrador de sensores: placa del *PCB*, soldadura y armado con las primera pruebas del equipo.
- Desarrollo de la aplicación web: Integración de la aplicación y el sistema *NMS*

4.2.1. Sistema NMS

La solución del sistema *NMS* se encuentra basada en *Zabbix*, una plataforma que dispone de amplias ventajas por ser un software moderno, intuitivo, gratuito y no menos importante, muy difundido en la comunidad. Pretende ser una solución flexible e híbrida entre tecnologías *on premise* y *cloud* (Fig. 10)

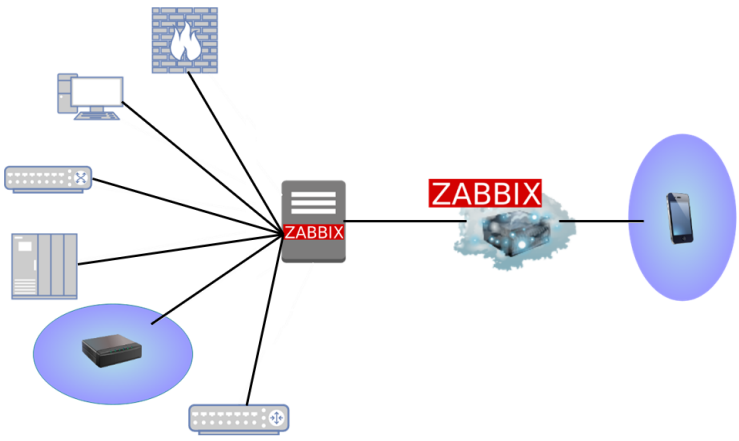

Figura 10: Mapa topológico de solución Zabbix

Los dos componentes fundamentales:

- Servidor *Zabbix*: Es un servidor *cloud*. Su función es centralizar los eventos de cada uno de los equipos de los sitios del cliente. Permite dar acceso a los usuarios desde cualquier lugar que se encuentren accediendo a su interfaz GUI con la IP pública del servidor o desde la aplicación móvil que se detallará más adelante.

- Proxy *Zabbix*: es un servidor *on premise*. El mismo establece un canal hacia el servidor *Zabbix* con un protocolo propio y, opcionalmente, encriptado. Gestiona la conexión de todos los equipos que dispone cada sede o sitio del cliente que soporte una conexión *SNMP* para reenviarla a este servidor. Parte de sus funciones es realizar la traducción de los OID del cliente con los *MIB*s, aligerando el procesamiento en el servidor *Zabbix* como así también el ancho de banda de cada de la sede correspondiente en la que se encuentre este Proxy. El Proxy *Zabbix* es una versión simplificada servidor *Zabbix* sin interfaz gráfica y con los complementos necesarios para cumplir su función. Se puede instalar en una plataforma de máquinas virtuales a disponer por el cliente. También, parte la solución de este proyecto y para independizar al cliente de disponer de estos servidores, se ofrece como solución alternativa una Raspberry Phi con el Proxy *Zabbix* instalado y configurado.

Las otras partes que conforman el esquema:

- Concentrador *SNMP* detallada en la sección 4.2.2
- Aplicación móvil detallada en la sección 4.2.3
- Otros equipos de red conectados por protocolo *SNMP*

Generalizando, este sistema tendrá un servidor *Zabbix* en la nube y múltiples servidores Proxy *Zabbix*, uno en cada sitio. Esto es una solución adecuada cuando el cliente no cuenta con cada una de sus redes interconectadas y centralizadas. Ante la caída de uno de los sitios no se pierde conexión con el servidor central como así tampoco con el resto de los sitios. También permitirá configurar notificaciones por mail, a la aplicación móvil, en la interfaz GUI, entre otras para cada evento que surja en los equipos. La interfaz GUI del servidor *Zabbix* presenta una pantalla de Login a través de la cual accederemos a la plataforma para su monitoreo y gestión. Comenzando con la parte de monitoreo tendremos un Dashboard (Fig. 11) estándar el cuál podemos modificar, eliminar e incluso crear otros. Entre las opciones se encuentra la visualización general de la cantidad de eventos sin resolver, una cronología de estos, gráficos de interés, mapas de red globales o de cada sitio, entre otras opciones.

Figura 11: Pantalla de Dashboard

Poseemos también una pantalla de problemas (Fig. 12) en la que podemos aplicar un filtrado específico y avanzado para buscar los eventos que deseemos.

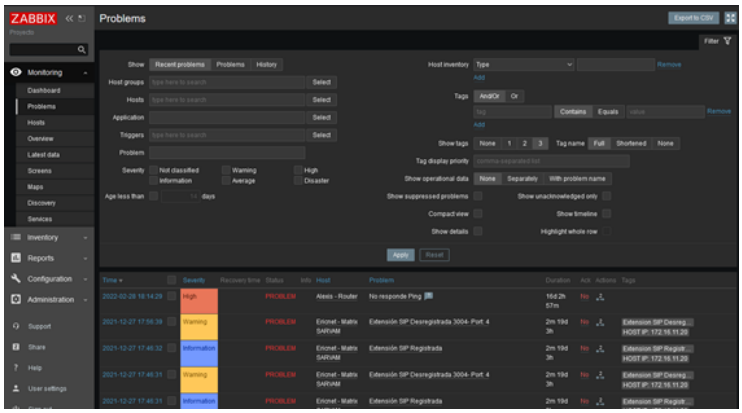

Figura 12: Pantalla de eventos

En la pantalla de hosts (Fig. 13) están todos los equipos que fueron conectados por *SNMP*. Muestra su IP, estado, conteo de eventos y demás.

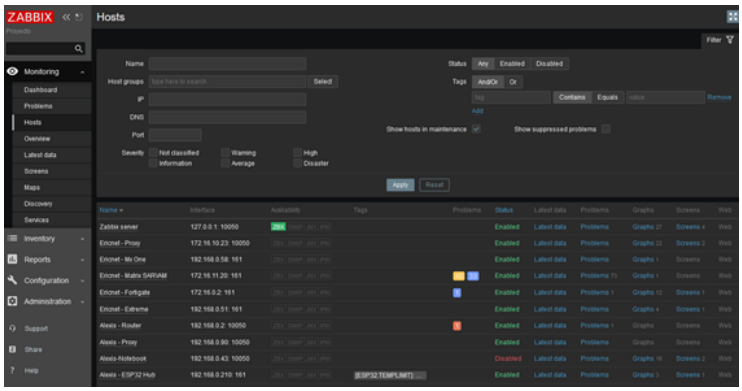

Figura 13: Pantalla de equipos

Finalmente tenemos la posibilidad de crear mapas (Fig. 14) para tener una percepción lógica de conexión como una rápida visualización de los equipos que se encuentren alarmados.

Dentro del apartado de configuración (Fig. 15) podremos realizar las configuraciones que permitirán la monitorización como así sus correspondientes. En la sección de configuración tendremos la posibilidad de agregar los equipos que tenemos por cada proxy, plantillas que nos permitirán crear funciones para los eventos y distintos comportamientos de nuestro *NMS*, acciones que permitirán hacer cambios (paquete tipo *SET* de *SNMP*) en nuestros equipos de acuerdo con las posibilidades de configuración que nos ofrezca el equipo remoto.

Si bien estás son las opciones de personalización general, existen otras que se podrán utilizar de acuerdo con las exigencias y requerimientos de nuestro cliente.

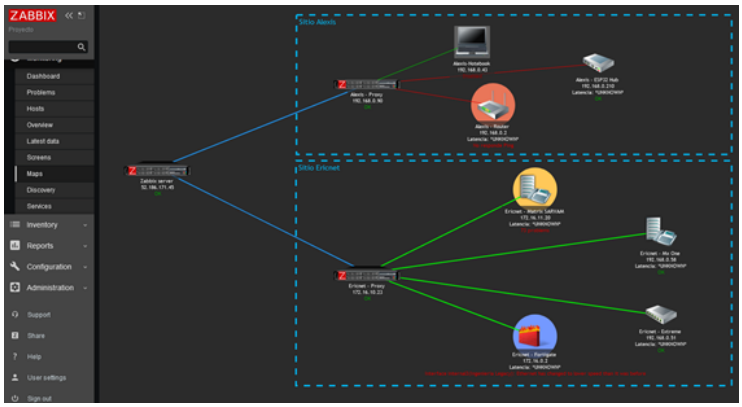

Figura 14: Pantalla de mapas

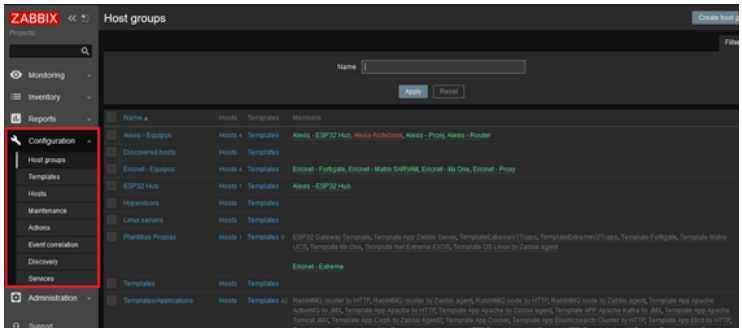

Figura 15: Sección de Configuración

4.2.2. Concentrador de sensores

El concentrador de sensores es un dispositivo desarrollado por los integrantes de este proyecto con la finalidad de completar la solución con la adición de un componente interesante. Como se ha mencionado anteriormente, el ambiente de *Datacenter* representa un espacio con equipos de gran interés que permiten a la instalación funcionar. Nuestro objetivo es tener control de todos ellos pero no tenemos control del espacio mismo como ser una falla de refrigeración o un acceso no autorizado a este espacio. Por lo tanto, la finalidad del concentrador de sensores es proporcionar la posibilidad de medir temperatura dentro de un rack, de la sala y controlar el estado de las puertas de los racks y la de acceso a la sala misma. La comunicación de este dispositivo con el servidor o proxy *Zabbix* lo realiza a través del protocolo *SNMP*, con el uso de mensajes *GET* y *SET*. Se desarrolló una interfaz HTTPS para la configuración inicial de la dirección IP, máscara de subred y la puerta de enlace predeterminada además de la visualización de los estados de los sensores. El diseño de la placa (Fig. 16) tiene 4 conectores para sensores de puerta magnéticos y un conector para el sensor de temperatura DS18B20.

Del otro lado de la placa se encuentra un conector RJ45 (Fig. 17) para conectarse por red y enlazarse con el servidor o proxy *Zabbix*.

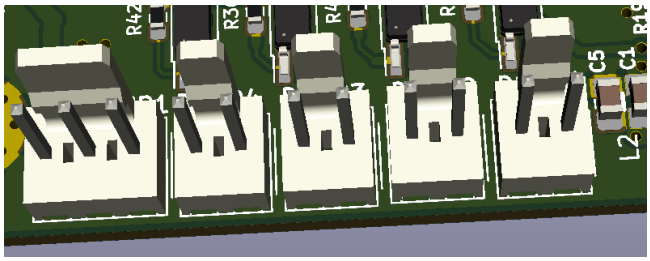

Figura 16: Conectores del Concentrador de Sensores.

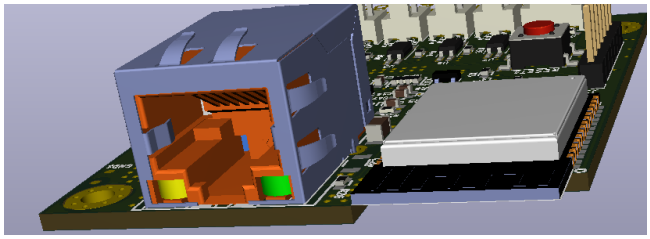

Figura 17: Conector RJ45 para la conexión Ethernet.

También, en uno de los laterales, cuenta con un conector Micro USB (Fig. 18) para alimentar el circuito con un adaptador USB a 5V estándar.

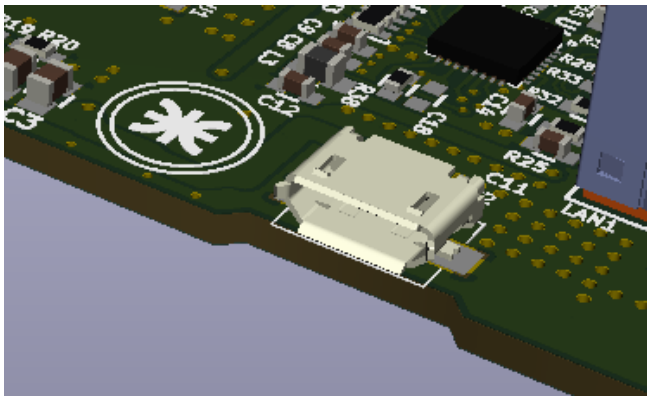

Figura 18: Conector Micro USB para alimentar el dispositivo.

En el otro lateral cuenta con los pines y el botón para la programación y debug de la misma (Fig. 19).

Finalizando, se puede observar (Fig. 20) en el centro de la placa el chip de Ethernet LAN8710 con todos los elementos pasivos asociados y en la esquina superior izquierda el microcontrolador ESP32 utilizado.

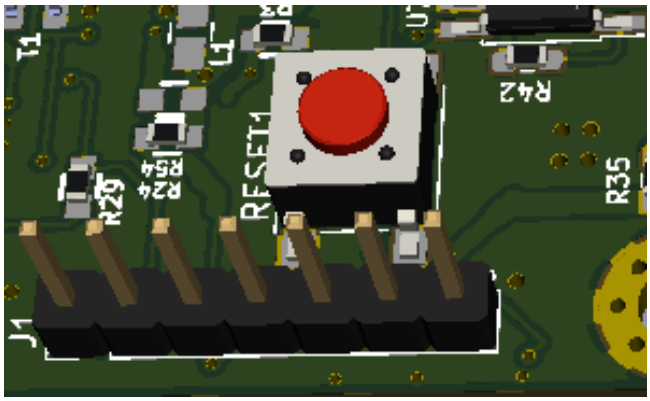

Figura 19: Conectores para la programación y debug de la placa.

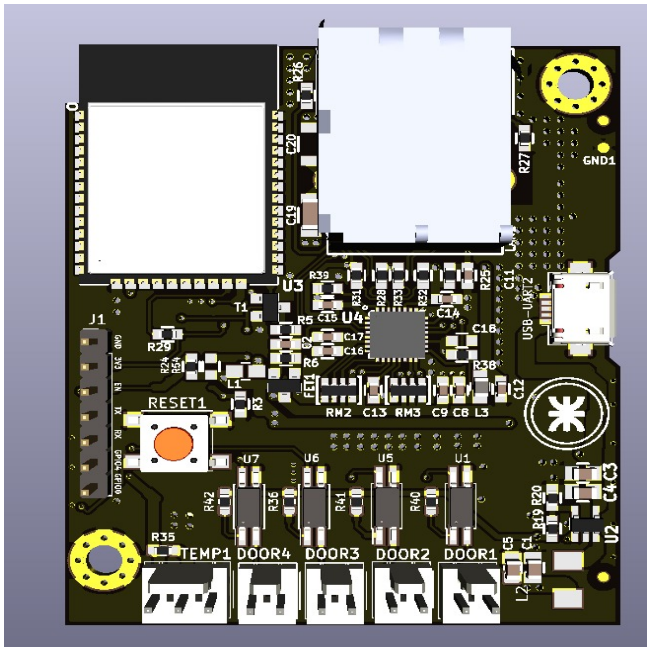

Figura 20: Placa completa.

4.2.3. Aplicación Móvil

En cuanto a la aplicación, se la diseñó de manera tal de que se encuentre modularizada mediante el uso de *Fragments*, usando solo tres *Activities* durante toda la aplicación y otorgando de esta manera una mayor flexibilidad.

Para empezar se tiene una Splash Screen (Fig. 21 izquierda) , que muestra el logo del *NMS* utilizado por tres segundos mientras se carga la aplicación.

Luego de esto, pasamos a la pantalla de login (Fig. 21 derecha), en la cual se carga la dirección IP, el nombre de usuario y la contraseña para poder ingresar. Cabe destacar que la aplicación recuerda el último inicio de sesión y autocom-

pleta los campos la próxima vez que se inicie la misma.

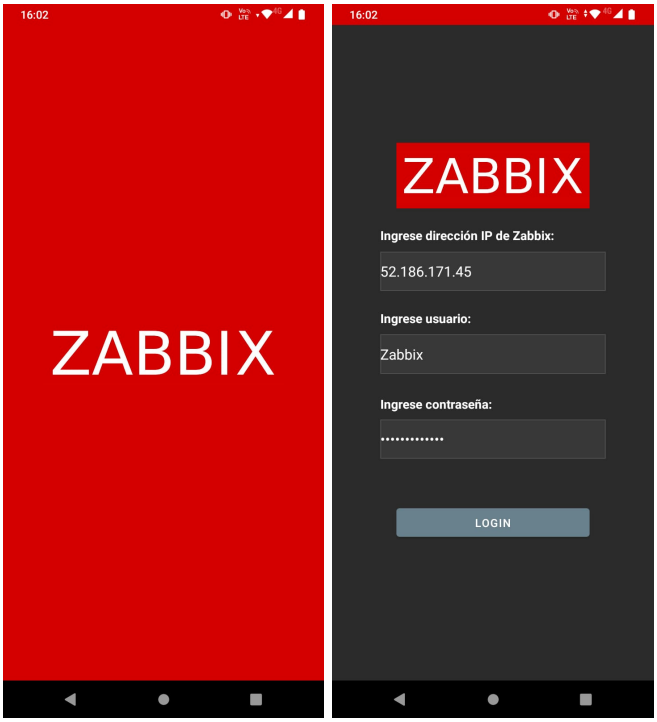

Figura 21: Izquierda: Splash Screen. Derecha: pantalla de login.

Luego de ingresar sesión se aprecia el menú principal, desde el cuál se puede cerrar sesión desde la *Toolbar* (Fig. 22 izquierda), y solicitar el alta o la baja del servicio de notificaciones desde un menú contextual en la Toolbar (Fig. 22 centro). Si se elige cerrar sesión, se vuelve a la pantalla anterior. Si se elige solicitar alguna acción de las notificaciones se abre un Dialog para confirmar la decisión (Fig. 22 derecha), y se utiliza el servicio de correo registrado en el dispositivo del usuario para poder mandar un mail haciendo la solicitud. Una vez terminado esto, se vuelve al menú principal. De esta manera un administrador agrega el token generado por el dispositivo y habilita el servicio de notificaciones mediante el uso de *Firebase Cloud Messaging* (Fig. 23).

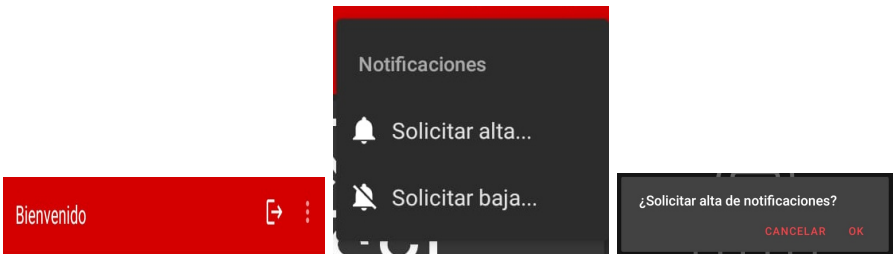

Figura 22: Izquierda: Barra del menu principal. Medio: Acciones de notificación. Derecha: Confirmación de notificación.

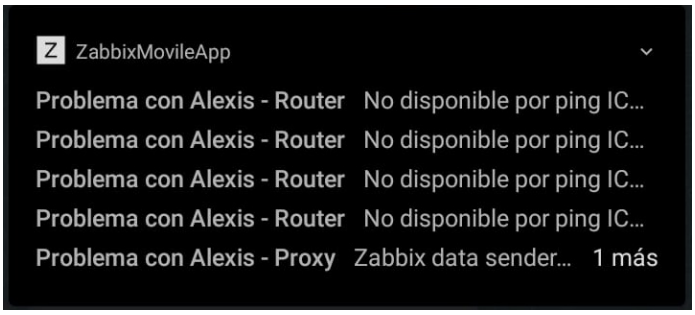

Figura 23: Notificaciones vistas desde el dispositivo del usuario.

Como última funcionalidad del menú se puede ingresar a las tres secciones de la aplicación (Fig. 24 izquierda): un tablero customizable en el cual se ven los ítems que el usuario considere importante monitorizar, la lista completa de hosts y sus respectivos ítems junto con la última medición de los mismos, y un listado de eventos histórico.

En la sección de eventos (Fig. 24 centro) aparece el listado de los mismos de los últimos 30 días, ordenados con su respectivo código de color de acuerdo a su severidad: sin categoría, información, de alerta, de riesgo promedio, de alto riesgo y desastrosos. Si uno lo desea puede filtrar los eventos por el host en el cuál se producen (Fig. 24 derecha).

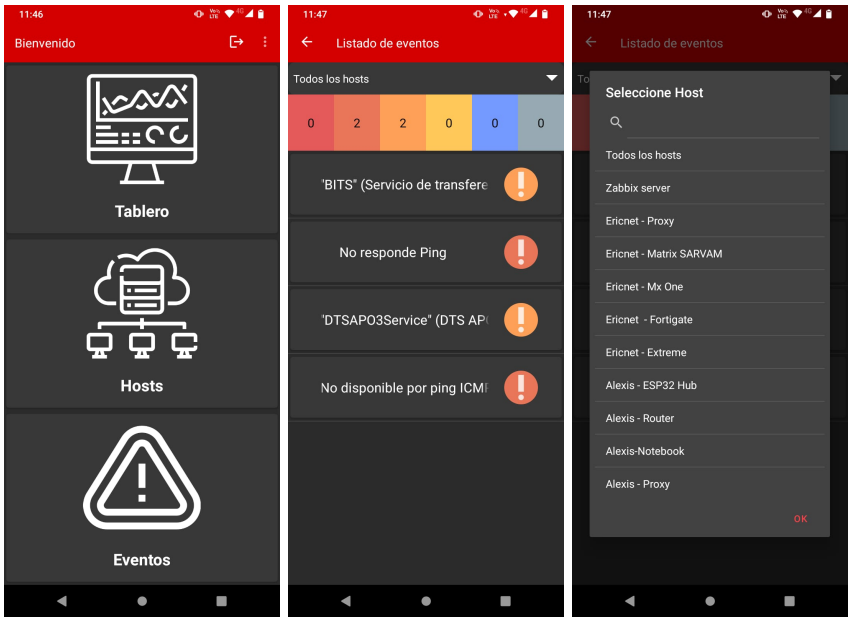

Figura 24: Izquierda: Menú principal. Centro: Menú de eventos. Derecha: Filtro de eventos por host

Al seleccionar alguno de los eventos, aparece su nivel de severidad, el momento en que se detectó y si está atendido o no (Fig. 25).

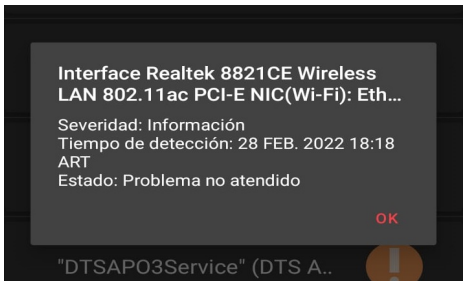

Figura 25: Selección de evento.

En la sección de hosts (Fig. 26 izquierda) aparece primero un listado de hosts que tiene disponible para visualizar el usuario, si se selecciona uno aparece un listado de grupos de ítems del host seleccionado (Fig. 26 centro) y si a continuación se selecciona alguno de estos grupos aparece un listado de todos los ítems pertenecientes a ese grupo con su último valor y la fecha y hora en el cual se midió el mismo(Fig. 26 derecha). Si se selecciona alguno de estos ítems aparecerá una descripción si es que tiene alguna (Fig. 27).

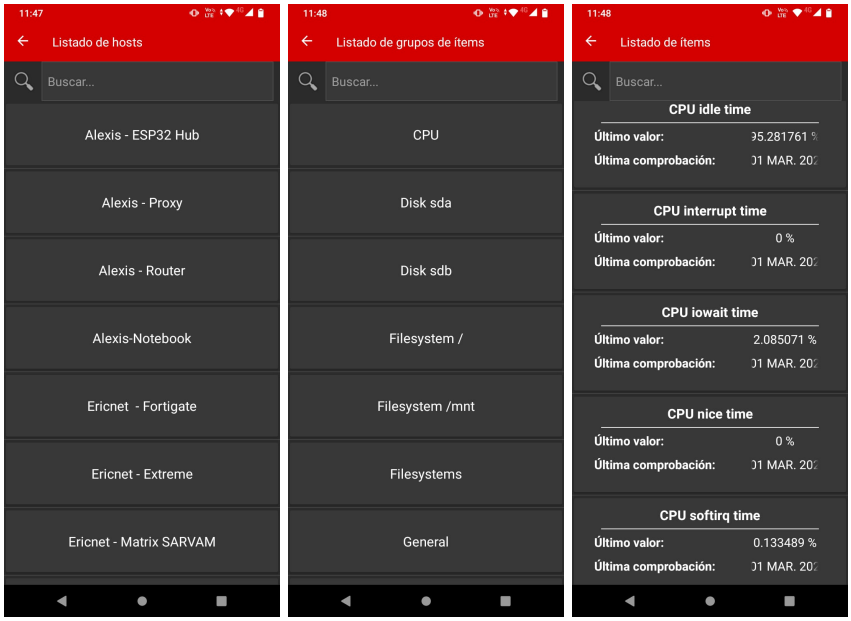

Figura 26: Izquierda: Menú de hosts. Medio: Menú de grupos de ítems. Derecha: Menú de ítems.

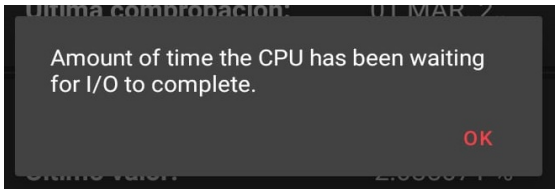

Figura 27: Selección de ítem.

Finalmente en la sección de tablero (Fig. 28 izquierda) aparecerá un listado de ítems guardados de manera local en el dispositivo del usuario de manera tal que no tenga que agregar un nuevo ítem cada vez que quiera monitorizarlo. Si se quiere agregar alguno primero se deberá seleccionar un host en el menú desplegable correspondiente, y un ítem en el otro menú desplegable. Si se selecciona algún ítem en el tablero, se mostrará una gráfica (Fig. 28 centro) con una ventana de tiempo modificable (Fig. 28 derecha). De la lista de la sección tablero se pueden agregar y quitar ítems (Fig. 29).

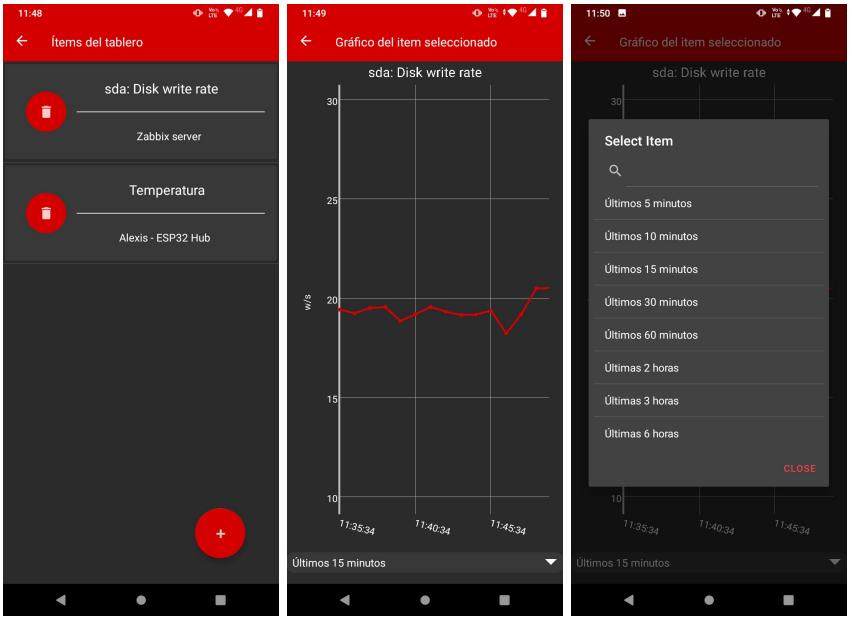

Figura 28: Izquierda: Menú del tablero. Medio: Gráfica de ítem. Derecha: Elección de ventana de tiempo.

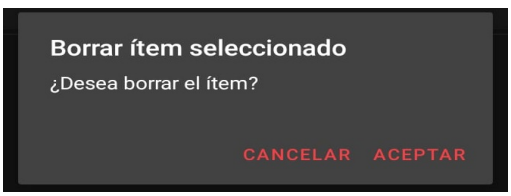

Figura 29: Confirmación de borrado de ítem de tablero.

5. Resultados, Mediciones y Verificación

Para la implementación de la solución propuesta se establece el siguiente ambiente de trabajo: Un Servidor *Zabbix* instalado y configurado en una maquina virtual de Azure. Un Proxy *Zabbix* instalado y configurado en una Raspberry Pi 4 conectado en la red local. Una red local para conectar el Proxy *Zabbix* y el concentrador de sensores. No hace falta abrir ningún puerto en el Router. Una PC con un navegador web para visualizar los gráficos provistos por *Zabbix* y un emulador de Android corriendo la aplicación móvil

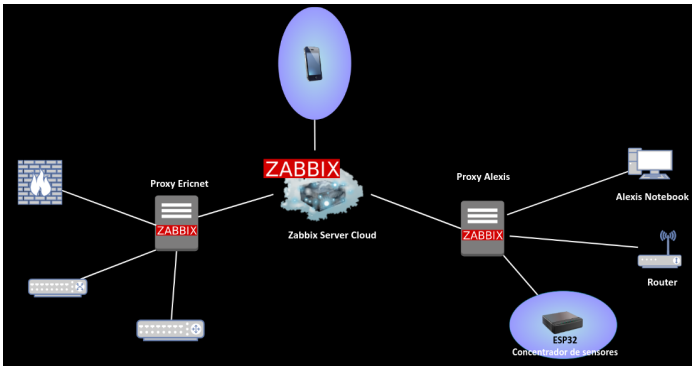

Figura 30: Ambiente de prueba

Manipularemos los sensores con el fin de observar en el sistema de monitoreo y su aplicación asociada los resultados. Esta serie de pruebas consta de:

- Sensor de puerta: manipular la puerta del gabinete de PC donde se encuentra el sensor de apertura de puerta.
- Sensor de Temperatura: cambios bruscos de temperatura.
- Se interactúa con la aplicación del sistema para ver los resultados de la creación de gráficos para monitorizar y configuración de notificaciones según los eventos de interés.

La descripción del procedimiento llevado a cabo se encuentra mejor detallada en el informe del Test de Aceptación, el cual forma parte de los anexos de este informe.

Realizando los procedimientos detallados anteriormente se obtuvieron los siguientes resultados:

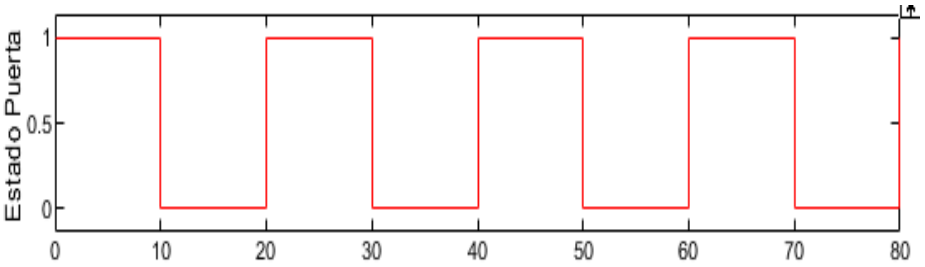

Figura 31: Excitación aproximada de la puerta, se abre y se cierra cada 20 segundos

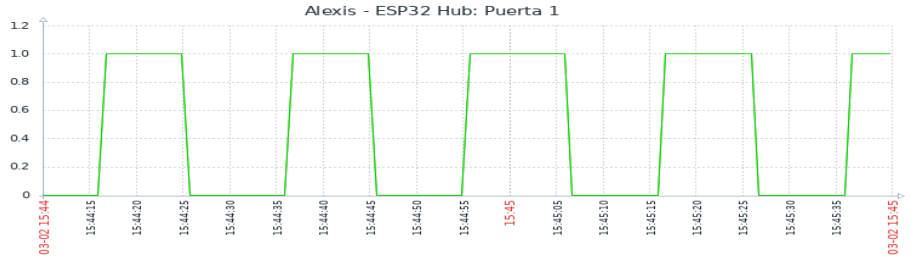

Figura 32: Respuesta de la puerta a la excitación

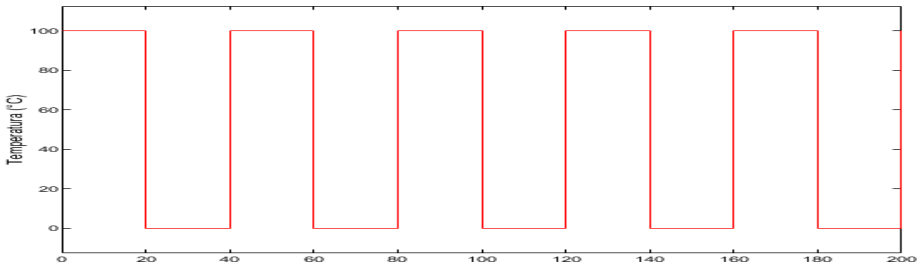

Figura 33: Excitación aproximada del sensor de temperatura, se lo coloca y se lo saca de un vaso con agua caliente

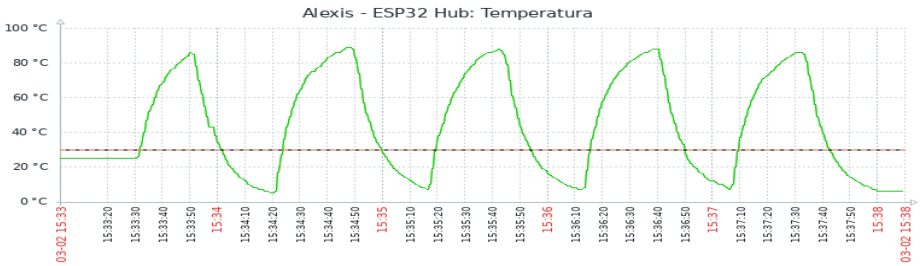

Figura 34: Respuesta del sensor de temperatura a la excitación

6. Conclusiones

Las pruebas realizadas al concentrador de sensores diseñado comprueban su correcto y esperado funcionamiento. En el caso de la topología de red, se logró instalar y configurar el servidor *Zabbix* en la nube de Azure, y el del proxy *Zabbix* en una Raspberry Pi, ambos de manera satisfactoria. Se concluye que los resultados obtenidos en el proyecto cumplen con lo propuesto inicialmente.

7. Referencias

https://www.zabbix.com/documentation/5.4/en/manual/concepts/proxy
https://www.zabbix.com/documentation/current/manual/api
https://www.jsonrpc.org/specification
https://www.unixtimestamp.com/

8. Índice de Figuras y Tablas

Índice de figuras

9. Apéndices

9.1. Circuitos eléctricos

Esquemático Concentrador

9.2. Hojas de Datos

Sensor de Temperatura: DS18B20
Sensor de Apertura de Puerta
Microcontrolador: ESP32 Wroom 32D

9.3. Programas Fuente

Repositorio de la aplicación Android
Repositorio del concentrador de sensores

9.4. Layout PCB

Repositorio con los Layouts

9.5. Otros Informes

Test de Aceptación
Póster
Fotos y Videos