International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2014): 5.611

Temperature Monitoring and Alert Generation System – An IoT Implementation

Shardul S. Thakre, Pratik S. Jain

Abstract: ***We present a prototype of an automated alert generation system in which there are plethora of sensors on which transmit temperature at regular intervals to an administrator who handles the equipments in case of any abnormalities and thereby taking necessary actions based on the alert generated and continuous monitoring of the device is possible. The data generated is stored into a database and can be analyzed using graphs generated.***

Keywords: Temperature sensor, LDAP authentication, MQTT, Apache Hbase, Service Gateway, SMTP Alerts, Jfree Chart graphs.

1. Introduction

Nowadays, the main communication form on the internet is human to human, but it foreseeable that in the near future that ant object will have a unique way of identification and can be addressed so that every object can be connected. This phenomenon is called Internet of Things – IoT.

In many industries, numerous temperature sensitive equipments are used, wherein it is essential to maintain the temperature at a certain threshold to avoid hazardous effects. Keeping a manual check on all the equipments at the same time is quite cumbersome. Here, our system can come handy.

In this paper, we present an autonomous system where temperature sensors are installed on all the temperature sensitive equipments and record the temperature at regular intervals. This value is sent further to a service gateway which processes the information to generate specific alerts based on the temperature sensed. With the help of such alerts, the administrator can take necessary actions to prevent a damage to the devices and equipments.

In our work, we have implemented a miniature model of the system proposed. The sensor is mounted on a small processing unit which is used to transmit the temperature values along with the sensor ID to the gateway. The gateway pushes this information into a database and form here, we can analyze the temperature based on our need and requirements.

2. Proposed System

This system operates in four stages:

In the First stage, the temperature sensor which is mounted on Raspberry Pi device records the current temperature and publishes the value in the form of MQTT message to the Service Gateway acting as an MQTT broker. The Raspberry Pi operates on a Debian Operating System and uses Paho C library functions for MQTT publishing. The message payload contains sensor ID and the temperature value. The message header contains username and password that would be used for LDAP authentication.

In the Second stage, the published message is received at the service gateway which acts as a MQTT broker. The Gateway first checks for authentication from LDAP server, which is supplied with username and password fields sent in the Basic Authentication header by Raspberry Pi. The message payload is logged into the Apache Hbase instance. This instance stores sensor ID and temperature in a single row, with sensor ID as row-key and temperature value as qualifier.

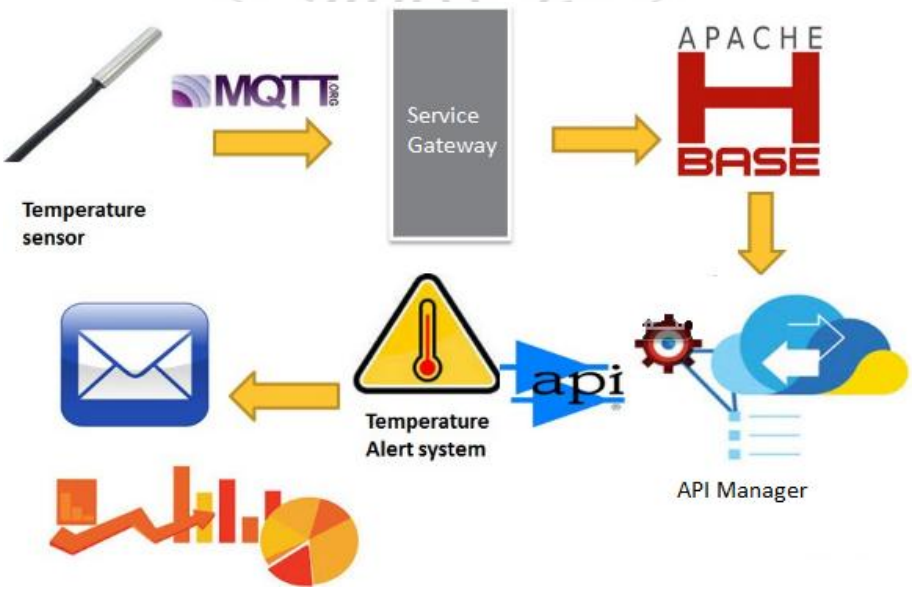

Figure 1: System Block Diagram

Figure 2: (a) Temperature sensor, (b) Raspberry Pi.

In the Third stage, the administrator can keep a check on the temperature within successive intervals by calling an API. The API management solution is provided by service gateway. Thus, when an API call is invoked, the service gateway scans the database and produces the values between the desired interval.

The fourth Stage consists of data analysis which has following parts:

1) SMTP Alert generation – Based on the temperature values retrieved from the database, an alert is generated in the form of email notification sent to the administrator for a particular threshold.
2) Graphical analysis – This is a plot of temperature values across different times. Thus, a discrepancy in the temperature n be identified and thermal analysis patterns can be derived.

3. Methodology

A. Temperature detection

Debian Operating system provides us with device drivers for various sensors. These can be implemented to obtain values from the sensors. Temperature values are recorded by the sensor and stored into a file. This file can be accessed and the value can be extracted.

```
/sys/bus/w1/devices/28-000005583a75/w1_slave
```

Above is the location where the temperature values are stored by the sensor. Further processing on the value is done to obtain a floating value up to 3 decimal places. The temperature thus received is sent to service gateway using *Paho* library function

```
MQTTClient_publishMessage(client, TOPIC, &pubmsg)
```

where *client* is the IP address of MQTT broker, *TOPIC* is the type of sensor – in our case "temperature", *&pubmsg* is a reference to the temperature value to be sent.

B. MQTT Protocol

MQTT (MQ Telemetry Transport) is a publish-subscribe based "light weight" messaging protocol for use on top of the TCP/IP protocol. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited. The publish-subscribe messaging pattern requires a message broker. The broker is responsible for distributing messages to interested clients based on the topic of a message.

MQTT Publish

After a MQTT client is connected to a broker, it can publish messages. MQTT has a topic-based filtering of the messages on the broker (check part 2 for more details on that), so each message must contain a topic, which will be used by the broker to forward the message to interested clients. Each message typically has a payload which contains the actual data to transmit in byte format.

Figure 3: MQTT Packet

The Quality of Service (*qos*) level is an agreement between sender and receiver of a message regarding the guarantees of delivering a message. There are 3 *qos* levels in MQTT: (i)At most once. (ii)At least once, (iii)Exactly once. In our case, we have used the second *qos* level.

C. Service Gateway

We have used a Service gateway, which acts as a broker for MQTT messages, as a tool to perform actions on database and also acts as an API management solution. The sensor is first authenticated using LDAP authentication technique to prevent any unauthorized entity to send data. The information thus received is then pushed into Apache Hbase database.

Further, this stored information, is made available through an API by the Service Gateway. Thus, when the administrator makes an API call, it is directed to the Service Gateway, which then runs a *scan* command on the database and returns values within a particular interval specified by the administrator.

D. SMTP Notifications

When the Service gateway detects an abnormal temperature with regard to a particular threshold value, it has to notify the administrator about this abnormality. This is done by sending an email alert to the administrator. For this purpose, we have implemented SMTP Server, which routes the notifications from Service gateway to the administrator's email account. The email generated consists of sensor ID, temperature and the intensity of alert based on the deviation from normal value.

From: <mqtt.gs.test@gmail.com>
Date: Aug 14, 2015 9:53 AM
Subject: URGENT: CLIENT NEED TO BE SHUTDOWN!
To: <mqtt.gs.test@gmail.com>
Cc:

The Temperature of following Client(s) is greater than the highest limit Please Handle Immidiately .....

[client487810]

Figure 4: Sample Alert

E. Graph Generation for Continuous Monitoring

While it can be quite onerous, to every time receive an email for every abnormality, we have devised a mechanism to generate graphs of real-time data, which displays patterns of the temperature values observed and thus any spike in the graph can be observed easily and thus necessary actions can be taken by the administrator before any substantial losses.

The graph generation is based on the sliding window phenomenon, i.e. the graph shows the value of temperature within a particular interval and refreshes after a fixed period of time.

Following graph shows the plot for sensor with id **487810.**

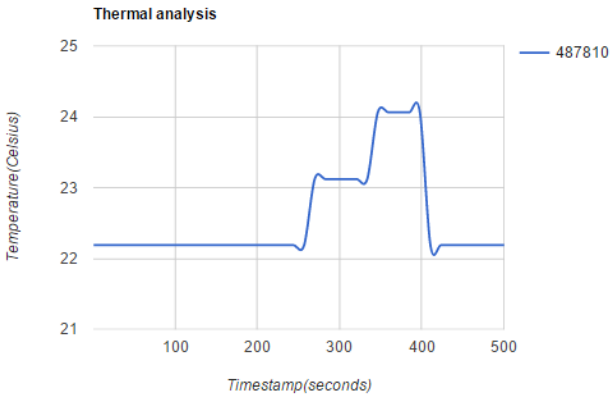

Figure 5: Sample Graph

4. Results and analysis

For the purpose of sending messages from the sensor, we have used a C program which implements Paho Client. When the sensor sends a message successfully, requests are seen on the service gateway side. Also the database reflects the same. Hbase provides us with storage for BigData, where our temperature values can be a record of very old data. Thus, we can store large amount of data. There are various alternatives for Hbase available in the market, which are much more efficient. We have implemented Hbase as it suited our need the most. Graphs were generated using Jfreechart library.

In this system, there are multiple remote points of failure. This has to be handled scrupulously. Any failures in sensor, can be verified from the device driver file of Debian OS. Network failure in MQTT transmission can be identified by power failure of sensor or gateway, or due to loss of connectivity. For large number of sensors, the service gateway can become a bottleneck and thus needs to be taken care of. Hbase failures are depicted on the service gateway. Data integrity of database is of prime importance. For that, proper security measures have to be taken.

We are also working on an Android application for graph generation and alert mechanism. Thus, the administrator can also monitor the activity even when he/she cannot be physically present in the office.

5. Conclusion

In this prototype, we have successfully implemented our proposed idea for a single sensor. But this system is scalable to a myriad extent. Thus, the entire process of monitoring can be automated thereby reducing the overhead of administrator. We are further working on this proposal to create a robust and more secure product.

Currently, we have considered only thermal analysis as our domain, but analogous systems can be used in varied domains, for instance, road accidents and health related issues can also be monitored.

6. Acknowledgment

We would like to express our gratitude to all those who helped us to complete this work. We want to thank our guide **Mr. Atahley P.** for his continuous help and generous assistance. He helped in a broad range of issues from giving us direction, helping to find the solutions, outlining the requirements and always having the time to see us.

We would like to thank our friend Mr. Magar P. for his help in providing us the raw material specifically the temperature sensor and Raspberry Pi device. We would like to thank our colleagues who helped us time to time from preparing report and giving good suggestions. We also extend sincere thanks to all the staff members of Department of Computer Science & Engineering for helping us in various aspects. Last but not least we are grateful to our parents for all their support and encouragement.

References

[1] Chen Zhou, Xiaoping Zhang “Toward the Internet of Things application and management: A practical approach”, World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium, 1-6, June 2014.
[2] Al-Fuqaha, A., Guizani, M. ; Mohammadi, M. ; Aledhari, M. ; Ayyash, M. “Internet of Things: A Survey on Enabling Technologies, Protocols and Applications”, Communications Surveys & Tutorials, IEEE (Volume:PP , Issue: 99), 1553-877X, 15 June 2015.
[3] De Caro, N., Colitti, W. ; Steenhaut, K. ; Mangino, G. ; Reali, G. “Comparison of two lightweight protocols for smartphone-based sensing”, Communications and Vehicular Technology in the Benelux (SCVT), 2013 IEEE 20th Symposium, 1 – 6, 21-21 Nov. 2013.
[4] Bandyopadhyay, S, Bhattacharyya, A.,” Lightweight Internet protocols for web enablement of sensors using

constrained gateway devices", Computing, Networking and Communications (ICNC), 2013 International Conference ,334 – 340, 28-31 Jan. 2013.

[5] Koutsonikola, V., Vakali, A., "LDAP: framework, practices, and trends", Internet Computing, IEEE (Volume:8 , Issue: 5), 66 – 72, 27 September 2004.

Author Profile

Shardul S. Thakre, Department of Computer Science and engineering, Shri Guru Gobind Singhji Institute of Engineering, and Technology, Nanded, India.

Pratik S Jain, Department of Computer Engineering, Maharashtra Institute Of Technology, Pune, Maharashtra, India.