Simple IoT simulation

Arvind Vishnubhatla

Abstract*: While analyzing iot projects it is very expensive to buy a lot of sensors , corresponding processor boards, power supplies etc. Moreover the entire process is to be replicated to cater to large topologies. The whole experiment is to be planned at a large scale before we can actually start to see analytics working. At a smaller scale this can be implemented as a simulation program in linux where the sensor data is created using a random number generator and scaled appropriately for each type of sensor to mimic representative data. This is them encrypted before sending it over the network to the edge nodes. At the server a socket stream now continuously awaits sensor data Here the required sensor data is retrieved and decrypted to give true time series data. This time series is now given to an analytics engine which calculates the trends and cyclicity and is used to train a neural network. The anomalies so found are properly deciphered. The multiplicity of the nodes can be characterized by having several client programs running in separate terminals. A simple client server architecture is thus able to simulate a large iot infrastructure and is able to perform analytics on a scaled model

KeywordsSimulate,Sensors,IOT,encryption,decryption,Linux, Client,Server,Analytics

I. INTRODUCTION

While analyzing iot projects it is very expensive to buy a lot of sensors, corresponding processor boards, power supplies etc[1]. Moreover the entire process is to be replicated to cater to large topologies. The whole experiment is to be planned at a large scale before we can actually start to see analytics working [2]. At a smaller scale this can be implemented as a simulation program in linux where the sensor data is created using a random number generator and scaled appropriately for each type of sensor to mimic representative data[3]. This is them encrypted before sending it over the network to the edge nodes. At the server a socket stream now continuously awaits sensor data . Here the required sensor data is retrieved and decrypted to give true time series data[4]. This time series is now given to an analytics engine which calculates the trends and cyclicity and is used to train a neural network [5]. The anomalies so found are properly deciphered [6]. The multiplicity of the nodes can be characterized by having several client programs running in separate terminals[7].A simple client server architecture is thus able to simulate a large iot infrastructure and is able to perform analytics on a scaled model[8].

II. PROJECT METHODOLOGY

A. Algorithm(Sensor Modelling)

The soil dielectric is measured through a capacitance probe ,where the dielectric permittivity of the soil is measured. Here we assume that the change in capacitance is available in the range 0-25 mv. The gyro measures the angular rate measured in degrees/second. Here we take the range to be 0-360 degrees respectively. The humidity is measured using a hygrometer in the range 0-100%. Light is measured using an LDR and varies from several thousands of ohms to indicate darkness to a few hundreds of ohms to indicate the presence of light. The magnetometer is used to measure the magnetic field using a gaussmeter in the range 0-3000 picotesla .The accelerometer is used to get orientation in the range +-1g to +-250g.The potentiometer gives data in the range 9mm to 1000mm. The pressure is measured from 4-20ma.The resolver gives data in the range 1 to 26VAC. The proximity sensor gives data in the range 0-2.5m.The temperature sensor gives data from 25 degrees to 150 degress Fahrenheit. The force sensor gives data from 0-100mm.The PIR sensor is used to detect infrared radiation . It goes high when the motion is detected for a specific period. The inductive sensor gives readings from 0-60mm.

B. Block Diagram

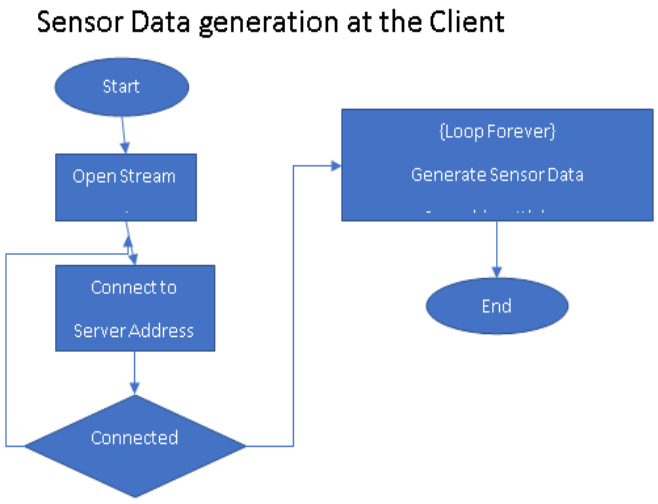

Figure 1 The client program flow in Linux

Revised Manuscript Received on November 10, 2019.
***** Correspondence Author
Arvind Vishnubhatla*,ECE,Gokaraju Rangaraju Institute of engineering and technology,Hyderabad,India,vainfo66@gmail.com

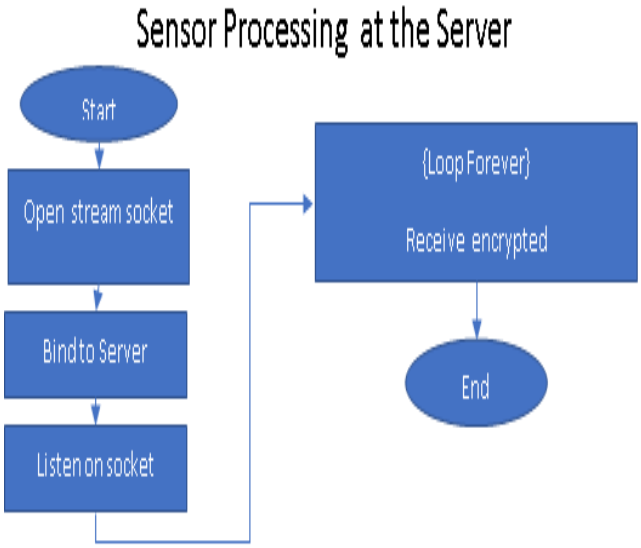

Figure 2 The Server program flow in Linux

C. Implementation

Two C programs tcpClient.c and tcpServer.c are written in Linux.

For compilation we use the commands gcc -c tcpClient.c tcpServer.c to get object files.

For linking we use commands gcc -o client tcpClient.o gcc -o server tcpServer.o

to get executables **client** and **server** in bold. In one of the windows the server is first run as ./server.Now several clients are executed from different linux terminals using ./client.. Log files are used to see if time series trends and analytics are successfully generated.

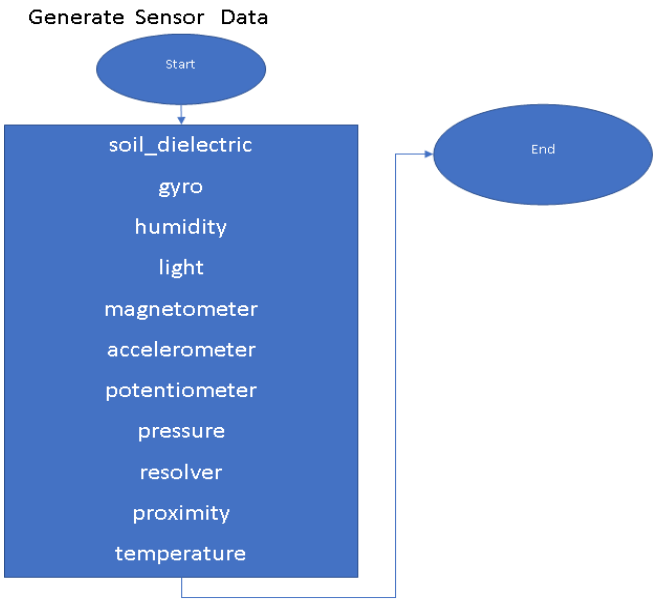

Figure 3 Generate sensor Data

A set of interconnected sensors ia attached to an embedded board used for industrial applications like surveillance ,automation etc.

A sensor detects changes in ambient conditions and converts these to electrical signals. A sensor node contains the power supply, processor unit ,sensing unit and a transceiver. Processing data at the sensor node requires high real estate and large processing power. Here network congestions due to multi-hop transmissions have not been modelled. The analytics is assumed to be done at the edge node.

III. RESULT ANALYSIS

The server program is first started from one of the linux windows using the command ./server. This shows up the message Server Socket is created,Bind to port 4444.
Listening…..
Now the client program is started from a separate window Using the command ./client.
The sensor data is now generated , encrypted and send to the Server. Appropriate instrumentation at the server shows that the data is received, unencrypted and printed. This is then sent to a time series module for analytics.
All sensor values are now plotted to generate the required dataset.
Now load testing of the module is performed by having several clients sending data to the server. Results show
That load testing runs satisfactorily.

IV. CONCLUSION

A sample implementation of the client server has been made by writing C programs for the client and server in Linux. The sensor processing is implemented using specific functions written in C. The execution results show that the simulator has been properly implemented.

REFERENCES

1. Benatallah, B.; Casati, F.; Toumani, F. (2004). "Web service conversation modeling: A cornerstone for e-business automation". *IEEE Internet Computing*. **8**: 46–54. doi:10.1109/MIC.2004.1260703.
2. Dustdar, S.; Schreiner, W. (2005). "A survey on web services composition" (PDF). *International Journal of Web and Grid Services*. **1**: 1. CiteSeerX 10.1.1.139.4827. doi:10.1504/IJWGS.2005.007545
3. Harper, Douglas. "server". *Online Etymology Dictionary*. Retrieved 30 November 2013.
4. Tolia, Niraj; Andersen, David G.; Satyanarayanan, M. (March 2006). "Quantifying Interactive User Experience on Thin Clients"
5. Otey, Michael (22 March 2011). "Is the Cloud Really Just the Return of Mainframe Computing?". *SQL Server Pro*. Penton Media. Retrieved 1 December 2013.
6. Barros, A. P.; Dumas, M. (2006). "The Rise of Web Service Ecosystems". *IT Professional*. **8** (5): 31. doi:10.1109/MITP.2006.123
7. Nieh, Jason; Yang, S. Jae; Novik, Naomi (2000). "A Comparison of Thin-Client Computing Architectures". *Academic Commons*. doi:10.7916/D8Z329VF. Retrieved 28 November 2018
8. "Distributed Application Architecture" (PDF). Sun Microsystem. Archived from the original (PDF) on 6 April 2011. Retrieved 2009-06-16.

AUTHORS PROFILE

Arvind Vishnubhatla
Areas of Interest: Signal Processing , Image Processing , Real time Embedded Systems , Photonics
Education

- Ph.D., Jawaharlal Nehru Technological University, Hyderabad
- MS, Northern Illinois University,Dekalb IL
- BE, Thapar institute of Engineering and Technology, PATIALA

Academic/Industrial Experience

1. Professor, ECE, Gokaraju Rangaraju institute of Engineering and Technology ,

2009-Present
2. Associate Professor,ECE, Gokaraju Rangaraju institute of Engineering and Technology , 2002-2009
3. Project Manager, Ericsson Communications India Limited**,** 2000-2002
4. Senior Software Engineer, **Motorola, Location (Network Solution Sector, Schaumburg, IL)**, 1999-2000
5. Senior Software Engineer, **Motorola, Location(Bangalore)**, India, 1993-1997
6. Software Engineer, **Bharat Heavy Electricals Ltd(Electronics Division. Bangalore)** 1989-1992