Raspberry Pi based Implementation of Internet of Things using Mobile Messaging Application - 'Telegram'

Vedang Ratan Vatsa
M.Tech Scholar
Dept. of Computer Science & Applications
M.D. University, India

Gopal Singh
Assistant Professor
Dept. of Computer Science & Applications
M.D. University, India

ABSTRACT

Home automation is turning out to be mainstream day by day because of its various points of interest. This can be accomplished either by local network administration or by using a remote control. This paper focuses on planning an essential home automation application using Raspberry Pi through perusing the content of the message sent from a cloud-based messaging mobile application - Telegram, the algorithm for it has been developed in the python environment which is also the default programming environment offered by Raspberry Pi. Results demonstrate the productive execution of proposed algorithm for home automation. Readings on a multimeter are used to demonstrate the switching activity in the GPIO pins of the Raspberry Pi.

Keywords

Raspberry Pi, Home Automation, Telegram, Python.

1. INTRODUCTION

Raspberry Pi (shown in the above figure) is a credit-card sized single-board computer unit that is developed in the UK with the goal of reaching to the people who cannot afford the mainstream computers available in the market and thus created a revolution in the field of personal computers.

The latest development of the Raspberry Pi mini- PC has opened great potential for the use of computing in a large number of fields. Because of the extraordinary points of interest of the Raspberry Pi framework, this innovation holds awesome guarantee for giving arrangements inside of the creating scene.

This incorporates however is not restricted to training devices, particularly the utilization of GPIO (General Purpose Input/Output) which permits robotized information procurement and creating basic computerized control frameworks in a school research facility environment. Most particular component of Raspberry Pi when utilized for instructive objects is GPIO module which consists of various pins, which permits interfacing with the broadly useful hardware.

This paper presents a basic application of raspberry Pi in the field of home automation and Internet of Things, in which the control signal of the respective GPIO pin of the Raspberry Pi is controlled by the content of the message received by the application client running on the Raspberry Pi.

The application used here is a cloud based multi-platform messaging application - 'Telegram'. The system responds to a particular message by triggering a set of instructions which may include a reply back to the sender and a python program which can further be used to take control of the GPIO pins on the Raspberry Pi.

The python program can be used to either make the control signal via GPIO high or low based on the requirements.

The Raspberry Pi can also be connected to a camera and a set of instructions can be triggered according to the utilization. In this way, pictures captured from the camera can also be sent over to the end user.

There can be numerous other applications based on this algorithm. The accepting of content from the end user by Raspberry Pi and replying according can also make it as a bot.

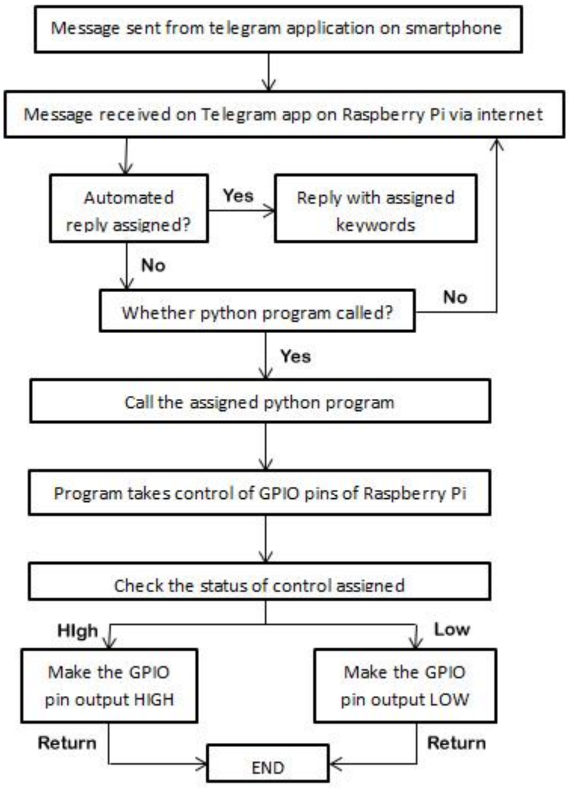
Thus the Raspberry Pi end of the client can also work as a bot to the user of the application so that a particular set of instructions can be set according to the content of the incoming message.

In this exceptionally creating period, where specifically or in a roundabout way, everything is reliant on processing and data innovation, Raspberry Pi ends up being a keen, monetary and productive stage for executing the home automation.

In this paper, we make the telegram application client on Raspberry Pi to accept a particular content as to trigger a python program which will further set the control signal to the respective GPIO pin of the board.

In order to evaluate the output and the performance of our algorithm we used a multimeter and attached it to the respective pins of Raspberry Pi, confirmed the application of the algorithm by getting the value of output voltage on the GPIO of Raspberry Pi via multimeter.

The figure shows the flow chart depicting the flow of events in the algorithm-


Message sent from telegram application on smartphone
Message received on Telegram app on Raspberry Pi via internet
Automated reply assigned?
Yes
Reply with assigned keywords
No
Whether python program called?
No
Yes
Call the assigned python program
Program takes control of GPIO pins of Raspberry Pi
Check the status of control assigned
High
Low
Make the GPIO pin output HIGH
Make the GPIO pin output LOW
Return
Return
END

2. METHODOLOGY

2.1 Setting up telegram

Telegram is based on accounts that partners or associates a client name with a mobile number. So as to set things up you truly need access to two mobile numbers, after that you just need one cellular telephone to utilize it. You can set up the Raspberry Pi to utilize the other cellular telephone number that you won't be utilizing. This second telephone is just truly expected to verify the record.

So you can arrange the username you wish to utilize, it is most likely a smart thought to introduce telegram on the cell telephone whose number is to be connected with the Raspberry Pi first. I utilized an iPhone and downloaded the application (free) from the App Store. It requests that you enter your telephone number (with nation code eg +91 for India) and it then sends a 5 digit verification code as a sms which you write into the application. On the other hand in the event that you don't get the code it can ring you after 2 minutes and give you the code with a programmed voice. It then approaches you for the client's First and Last name. Since I wasn't going to utilize telegram therefore on this telephone, I appointed my first name as Raspberry and last name as Pi. You can utilize whatever you wish.

Back to the Raspberry Pi now, the commands below can be utilized to successfully run the telegram client on Raspberry Pi

First of all, update the packages using the following commands:

“sudo apt-get update

[ENTER]

sudo apt-get upgrade

[ENTER]

Now in order to install the libraries: readline or libedit, openssl and (if you want to use config) libconfig and liblua.

sudo apt-get install libreadline-dev libconfig-dev libssl-dev lua5.2 liblua5.2-dev libevent-dev make

[ENTER]

Now in order to Clone GitHub Repository,

Git clone –recursive https://github.com/vysheng/tg.git && cd tg

[ENTER]

./configure

[ENTER]

make

[ENTER]

Running this command takes a little time and after the installation is done.

Now in order to navigate to the folder tg

cd tg

[ENTER]

then

bin/telegram-cli –k tg-server.pub –W –s action.lua

[ENTER]

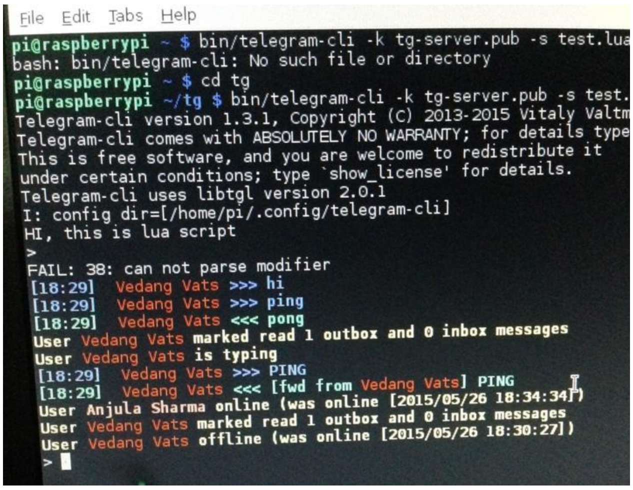
The first time we start telegram we must enter the phone number, including land code (for Italy is +91).

You should receive on your phone a sms message with a code, enter it and hit "Enter"

Now we are ready to use telegram, if you send a message from your smartphone you can see it on terminal.

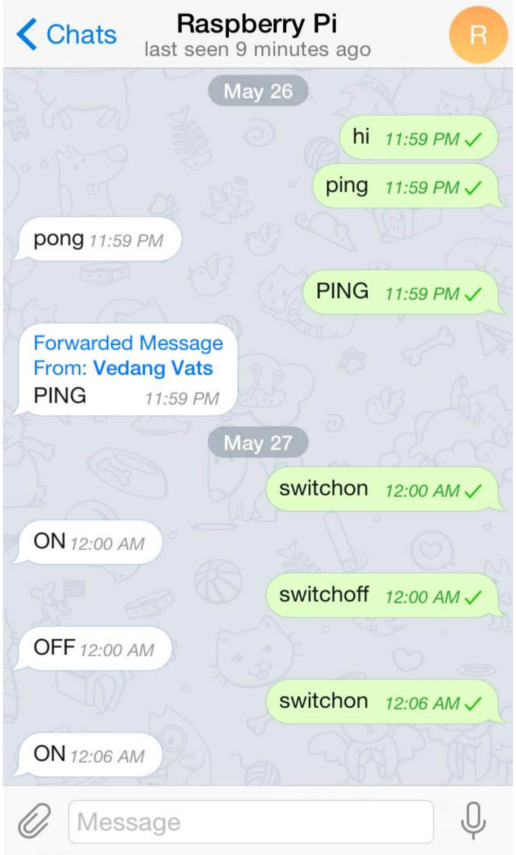
To send a message type

Msg Name_Lastname My message from Raspberry Pi”



```
pi@raspberrypi ~ $ bin/telegram-cli -k tg-server.pub -s test.lua
bash: bin/telegram-cli: No such file or directory
pi@raspberrypi ~ $ cd tg
pi@raspberrypi ~/tg $ bin/telegram-cli -k tg-server.pub -s test.
Telegram-cli version 1.3.1, Copyright (C) 2013-2015 Vitaly Valtm
Telegram-cli comes with ABSOLUTELY NO WARRANTY; for details type
This is free software, and you are welcome to redistribute it
under certain conditions; type `show_license' for details.
Telegram-cli uses libtgl version 2.0.1
I: config dir=[/home/pi/.config/telegram-cli]
HI, this is lua script
>
FAIL: 38: can not parse modifier
[18:29]  Vedang Vats >>> hi
[18:29]  Vedang Vats >>> ping
[18:29]  Vedang Vats <<< pong
User Vedang Vats marked read 1 outbox and 0 inbox messages
User Vedang Vats is typing
[18:29]  Vedang Vats >>> PING
[18:29]  Vedang Vats <<< [fwd from Vedang Vats] PING
User Anjula Sharma online (was online [2015/05/26 18:34:34])
User Vedang Vats marked read 1 outbox and 0 inbox messages
User Vedang Vats offline (was online [2015/05/26 18:30:27])
> 
```

The figure above shows the screenshot of the telegram client running on the Raspberry Pi



Raspberry Pi
last seen 9 minutes ago

May 26

hi 11:59 PM ✓

ping 11:59 PM ✓

pong 11:59 PM

PING 11:59 PM ✓

Forwarded Message
From: **Vedang Vats**
PING 11:59 PM

May 27

switchon 12:00 AM ✓

ON 12:00 AM

switchoff 12:00 AM ✓

OFF 12:00 AM

switchon 12:06 AM ✓

ON 12:06 AM

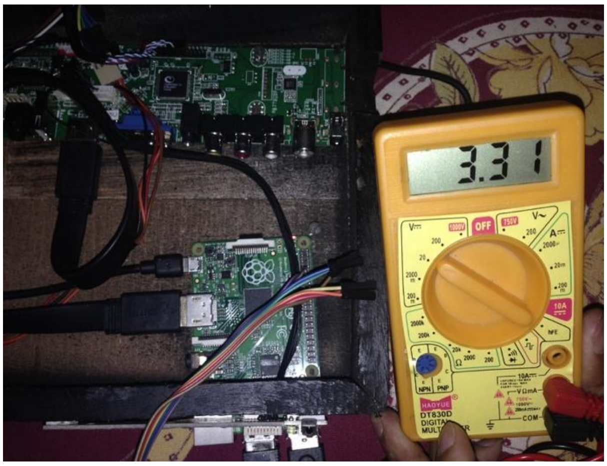
The above figure shows the chat session screenshot on a smartphone with the Telegram client on the raspberry Pi.

Now in order to control the GPIO pins of the Raspberry Pi, two python program files named switchon.py and switchoff.py are created which directs the control of the respective GPIO pin which in this case is 17 according to the BCM configuration. The python programs are called as the Telegram client in raspberry pi receives specified input switchon and switchoff as message and calls the respective python program files by the use of sudo command

3. PERFORMANCE EVALUATION

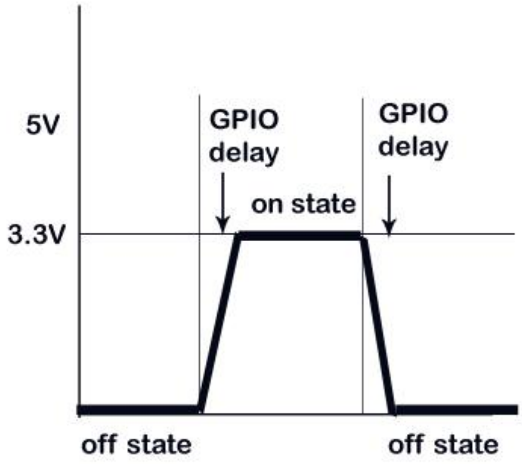
In order to evaluate the output and the performance of our algorithm we used a multimeter and attached it to the respective pins of Raspberry Pi. In our case we used the BCM pin configuration and used the GPIO pin 17 (BCM). So the positive pin of the multimeter is attached to the GPIO 17 pin and the negative pin of the multimeter is attached to the GND of Raspberry Pi.

As the *switchon* message is received by the Raspberry Pi Telegram client, it replies with a message stating ON back to the sender and on the other hand it also activates a python program named switchon.py which makes the GPIO pin 17 HIGH.



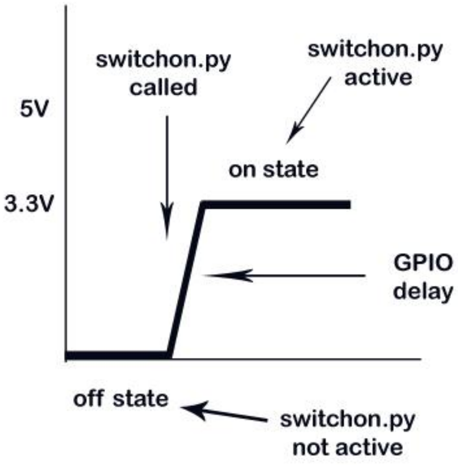
Similarly when the Raspberry Pi Telegram client receives *switchoff* message, it replies with a message stating OFF back to the sender and also activates a python program named switchoff.py which makes the GPIO pin 17 in LOW state.

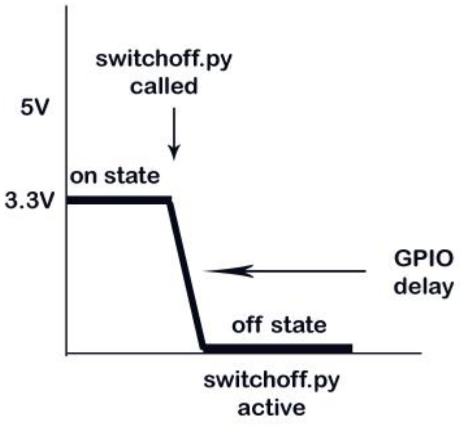
The following graph shows the two states, i.e., ON and OFF according to the called program. The graph also shows the GPIO delay which is the time taken by the program to activate the required pin which in our case is 17.



GPIO pin state graph

The following graphs shows the two states, i.e., ON and OFF as per the called switchon.py and switchoff.py programs respectively which activates the pin 17 according to the BCM configuration. The GPIO delay is the delay in between the called program and the pin activation and is showed clearly in the graph.





The above graph shows the change of state when switchon.py function is called while the graph below shows the change of states when switchoff.py is called.

Message sent from Telegram app on smartphone	Reply from Telegram Client on Raspberry Pi	Python program called	GPIO pin used	GPIO status
ping	pong	none	-	LOW-0
pong	ping	none	-	LOW-0
switchon	ON	switchon.py	17(BCM)	HIGH-1
switchoff	OFF	switchoff.py	17(BCM)	LOW-0
PING	Forwarded Message From : Vedang Vats PING	none	-	LOW-0

4. CONCLUSION & FUTURE SCOPE

Today everything is reliant on calculation and data innovation, Raspberry Pi turns out to be a savvy, monetary and effective stage for executing the home automation.

This paper gives an essential utilization of Raspberry Pi in home automation. The code used is very generic and adaptable in an easy to use way and can be stretched out for any future usage like power control mechanism, surveillance system, anti-theft system and so forth, effectively.

Also, this strategy is superior to other home automation techniques is a few ways. Since Telegram is a cloud based messaging application with focus on security and speed, it makes it easy to use by anyone since it is being offered in most of the platforms available in the market.

What makes it unique is the secure end-to-end user protocol that it makes use of and the speed which it provides. On the other hand, it offers the unique property of being heavily encrypted and self-destruction of messages.

5. REFERENCES

[1] Benisch, M., Kelley, P. G., Sadeh, N. & Cranor, L. F. Capturing location privacy preferences: quantifying accuracy and user-burden tradeofs. Personal Computing, in 15, (2011), 679-694.

[2] Berzowska, J. Memory rich clothing: second skin that communicate physical memory. In Proc. of ACM Press in (2005), 32-40.

[3] Brush, A. J. B., Lee, B., Mahajan, R., Agarwal, S., Saroiu, S. & Dixon, C. Home automation in wild: challenges and opportunities. In Proc. CHI 2011, ACM Press (2011), 2115-2125.

[4] Botterman, M. Internet of Things: an early reality of the Future Internet. Report of Internet of Things workshop. European Commission Information Society and Media Directorate General (2009) at Pragua.

[5] Carretero, J. & García, J. D. The Internet of Things: connecting the world. Personal Ubiquitous Computing (2013).

[6] Costanza, E., Panchard, J., Zufferey, G., Nembrini, J. Freudiger, J., Huang, J. & Hubaux, J. SensorTune: mobile

[7] auditory interface for the DIY wireless sensor networks. In Proc. CHI 2010, ACM Press (2010), 2317-2326.

[8] Denning, T., Borning, A., Friedman, B., Gill, B. T., Kohno, T. & Maisel, W. H. Patients, pacemakers, and implantable defibrillators: security for wireless implantable medical devices. In Proc. CHI 2010, ACM Press (2010), 917-926.

[9] Erickson, T., Li, M., Kim, Y., Deshpande, A., Sahu, S., Chao, T., Sukaviriya, P. & Naphade, M. The Dubuque electricity portal: evaluation of city-scale residential electricity consumption feedback system. In Proc. CHI 2013, ACM Press (2013), 1203-1212.

[10] Gaver, W. W., Bowers, J., Boehner, K., Boucher, A., Cameron, D. W. T, Hauenstein, M., Jarvis, N. & Pennington, S. Indoor weather stations: investigating a ludic approach to environmental HCI through batch prototyping. In Proc. CHI 2013, ACM Press (2013), 3451-3460.

[11] Grosse-Puppendahl, T., Braun, A., Kamieth, F. & Kuijper, A. Swiss-Cheese extended: an object recognition method for ubiquitous interfaces based on capacitive proximity sensing. In Proc. CHI 2013, ACM Press (2013), 1401-1410.

[12] Harrison, C., Tan, D. & Morris, D. Skinput: appropriating the body as an input surface. In Proc. CHI 2010, ACM Press (2010), 453-462.

[13] Hong, X. & Nugent, C. D. Segmenting sensor data for activity monitoring in smart environments. Personal Ubiquitous Computing, 17, (2013), 545-559.

[14] Ishii, H. Tangible bits: beyond pixels. Proc. of TEI, ACM Press (2008), xv-xxv.

[15] Jacobs, R., Benford, S., Selby, M., Golembewski, M., Price, D. & Giannachi, G. A conversation between trees: what data feels like in the forest. In Proc. CHI 2013, ACM Press (2013), 129-138.

[16] Koreshoff, T. L., Leong, T. W. & Robertson, T. Approaching a human-centred Internet of Things. In Proc. OZCHI (2013).

[17] Marquardt, N., Taylor, A. S., Villar, N. & Greenberg, S. Rethinking RFID: awareness and control for interaction with RFID systems. In Proc. CHI 2010, ACM Press (2010), 2307-2316.

[18] Patel, S. N., Gupta, S. & Reynolds, M. S. The design and evaluation of an end-user-deployable, whole house, contactless power consumption sensor. In Proc. CHI 2010, ACM Press (2010), 2471-2480.

[19] Pierce, J. & Paulos, E. Beyond energy monitors: interaction, energy, and emerging energy systems. In Proc. CHI 2012, ACM Press (2012), 665-674.

[20] Raij, A., Ghosh, A., Kumar, S. & Srivastava, M. Privacy risks emerging from the adoption of innocuous wearable sensors in the mobile environment. In Proc. CHI 2011, ACM Press (2011), 11-20.

[21] Reichl, P., Froehlich, P., Baillie, L., Schatz, R., & Dantcheva, A. The LiLiPUT prototype: a wearable lab environment for user tests of mobile telecommunication applications. In Proc. of CHI 2007, ACM Press (2007), 1833-1838.

[22] Saponas, T. S., Tan, D. S., Morris, D., Turner, J. & Landay, J. A. Making muscle-computer interfaces more practical. In Proc. CHI 2010, ACM Press (2010), 851-854.