International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-4, November 2019

Security System for Visitor Validation at Entrance using Raspberry Pi and Elliptical Curve Digital Signature

S Rajashree, Sukumar R, Bhuvankumar P

Abstract*: This paper proposes the design and development of security system for visitor validation at entrance using Raspberry Pi and Elliptical curve digital signature with WiFi network connectivity. When visitor presses the doorbell switch at entrance it is captured by the GPIO pin of Raspberry Pi. Once Raspberry Pi gets the signal it activates the Raspi camera to capture the image in jpg format. The image gets time stamped. The data in the jpg format is than authenticated using ECDSA (Elliptical Curve Digital Signature Algorithm) process that takes current time as seed to generate private key and required signatures. The authenticated image along with its signature are zipped and sent to the authorized e-mail address. Upon receiving the e-mail notification the image file can be verified by the user in both Windows and Linux operating system that has verification program installed in it to know whether signature matches with the image. The advantage of the system is that the user can verify image from anywhere and anytime and can respond to the situations. The setup uses Raspberry Pi 3, Raspberry Pi Camera and MIRACL software library, along with tactile switch and bread board.

Keywords: ECDSA, Smart door, Raspberry pi 3, Visitor validation

I. INTRODUCTION

In the present world to provide high security authentication of user entering a home or any private premises becomes more important as to avoid intruders entering the private area resulting in security issues. One way of achieving this would be to incorporate cameras near the entrance and look before opening the door. But this requires person who knows about all other visitors to be present all the time near the entrance. Another way is to have a system that has the capability to send authenticated information about the visitor to the owner and get back response from him weather that person can be allowed inside the premises or not. In this paper a similar system is designed using Smart device Raspberry Pi along with Raspi-pi camera. Raspberry pi being a microprocessor device has capability to collect and distribute the information from the users and surroundings to the owner.

Manuscript published on November 30, 2019.
* Correspondence Author
S Rajashree*, Research Scholar, Department of Computer Science, Jain University, Bangalore, India
Dr. Sukumar R, Professor, Department of Electronics and Communication, Jain University, Bangalore, India
Bhuvankumar P, Student, Department of Electrical and Electronics Engineering, KSSEM Bangalore

II. RELATED WORK

Security in IoT environment is realized through the following layered architecture[1]:

a. Perception layer: this includes devices like sensors camera etc.., which send raw data to smart devices.
b. Data processing layer: securing the data from the IoT end devices before transmitting it over the network.
c. Network layer: this includes devices like IoT gateway which transfer signed and received data from smart devices to the destination. Security using IP spoofing using source address validation is described in [2].
d. Application Layer: receive data from network decode the information and present it to the user in a format as desired by the user.

There are challenges in transmitting the user data over a public network due to security issues. Among the available security mechanisms for data transfer ECDSA is found to be more reliable and efficient[3]. The present mechanisms for security at the entrance (home and office) are not efficient as they transfer raw data from the users to the monitoring locations. The data can get manipulated when transferring over a public network there by defeating the purpose of security.

The existing mechanisms for providing home security include capturing the image of unauthorized person and feeding it to Raspberry pi device and sending it to the authorized e-mail for monitoring [4]. The image disadvantage is any person coming in contact to sensor is captured and sent which can involve large amount of processing and data transfer, data can also get manipulated and corrupted. Security using doorbell mechanism and data encryption through AES is discussed in [5]. Security using smart door lock and image recognition is discussed in [6], this requires maintenance of large amount of data.

III. SYSTEM DESIGN

The system design involves setting up of hardware components and installation of required packages (software requirement) as mentioned below.

a. Camera connection and settings

In the Raspberry Pi device install the following packages:

sudo apt – get update
sudo apt – get upgrade
sudo raspi – config

and select enable for camera.[7]

Install imagemagick[8] library package to write date and time on the image.

b. SMTP mail with mpack setting

sudo apt – get install ssmtp

$sudo\ apt - get\ install\ mailutils$

Edit the configuration file

$sudo\ nano\ /etc/ssmtp/ssmtp.conf$

Add the following

$root = postmaster$

$mailhub = smtp.gmail.com: 587$

$hostname = raspberrypi$

$AuthUser = AGmailUserName@gmail.com$

$AuthPass = TheGmailPassword$

$FromLineOverride = YES$

$UseSTARTTLS = YES$

For sending the attachments install mpack using following command

sudo apt – get install mpack [9]

c. MIRACL library settings

MIRACL is 'Multi precision Integer Rational Arithmetic Crypto Library'. MIRACL can be installed on Microsoft Visual Studio 2019 as well as smaller devices. MIRACL Cryptographic SDK: MULTIPRECISION INTEGER AND RATIONAL ARITHMETIC CRYPTOGRAPHIC LIBRARY is a C software library which is one of the standard open-source SDK for Elliptic Curve Cryptography (ECC).

Download MIRACL crypto library from *https://github.com/miracl/MIRACL* which is available in zip format and install using the command:

$unzip\ -j\ -aa\ -L\ MIRACL-master.zip$ [10]

Edit linux file copy appropriate file (*mrmuldv.ccc*) and run bash Linux command to create *miracl.a* library.

d. Hardware setup

Connecting the Raspberry Pi's general-purpose input output ports (GPIO) to a tactile switch

- We connect one side of the switch to an input pin on the Raspberry Pi using pin 18 [general purpose input output ports (GPIO)]
- The other side of the switch is connected to ground using pin 6 as shown in Figure 1 using jumper wires.
- when the button is pressed program runs that reads the state of the button.[11]

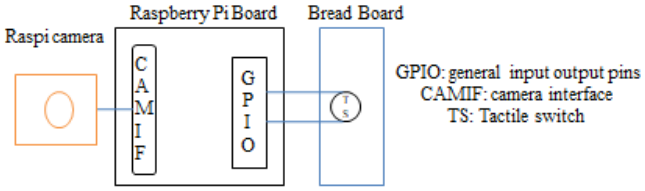

Fig.1. Raspberry pi to switch connection

Working of ECDSA algorithm

Authentication of data can be achieved using digital signatures. One of the efficient algorithms for digital signature is using Elliptical curve. Elliptic Curve Cryptography (ECC) is one of the most powerful cryptographic methods as it requires less computational effort. ECDSA is a mathematical scheme used for verifying the authenticity of digital file. ECDSA is a public-key cryptography which uses pairs of keys (public key distributed publically and private keys which is kept secret).

ECC protocols (ECDSA) is implemented using point arithmetic operations such as point doubling and point addition. The ECC curve equation is given by $y^2 = x^3 + Ax + b\ mod\ p$. This equation uses the variables like [3A,B] which is given by NIST p192 elliptical curves with prime field. The information about the parameters used in the equation (NIST p192) are given below-

$$p = FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFFFFFFFFFFFFFFFF$$

$$A = -3$$

$$B = 64210519E59C80E70FA7E9AB72243049FEB8DEECC146B9B1$$

$$n = FFFFFFFFFFFFFFFFFFFFFFFF99DEF836146BC9B1B4D22831$$

$$Gx = 188DA80EB03090F67CBF20EB43A18800F4FF0AFD82FF1012$$

$$Gy = 07192B95FFC8DA78631011ED6B24CDD573F977A11E794811$$

Elliptical curve Digital signature has in 3 phases:

- Phase 1: Generation of keys
- Phase 2: Signing of file(R, S)
- Phase 3: Verification of signed file

Phase 1: Generating of keys:

For generating the key ECDSA uses the base point G along with a large prime number (d_A)

$$Q_A = d_A * G \text{ ------1}$$

- "d_A" is the private key (random number) That is kept secret in the device
- "Q_a" is the public key that will be shared along with the signatures
- "G" is the base point defined by NIST P192

Phase 2: Creating signature:

The signature is 40 bytes and is represented by two values of 20 bytes each, the first one is called R and the second one is called S. so the pair (R, S) along with public key is shared with the actual data file.

- *Finding the value of R*

Generate a random value 'k'(of 20 byes) using random number generator or seed, and use point multiplication to calculate the point on the curve

$$P = k * G \text{ ----------2}$$

That point p 'x' co-ordinate value will represent 'R'.

- *Finding the value of S:*

$$S = K^{-1}(Z + d_A * R) mod\ P \text{ ------3}$$

Z is hash of the message that has to be signed

K^{-1} is the multiplicative inverse of K

R is the x co-ordinate value of point p found by using equation 2.

Phase 3: Verifying the sign

The original message will be sent along with public key (Q_A) and the signature (R, S). The verification process uses the message and finds the hash value of the message and computes the value p given by the equation 4. If both the value R and computed value are same then the message is accepted as original.

$$P = S^{-1}\ Z\ * G\ + S^{-1} * R * Q_A \text{----------4}$$

IV. IMPLEMENTATION

Home security system for visitor validation consists of capturing of visitor image, sending an authenticated image to the e-mail of the intended owner and verification of the image by the owner of the system to make proper decision. The implementation of this system involves the following steps:

Step 1: This involves the owner of the system to place Raspi camera in the location where a visitor who is pressing the switch is visible to the camera. When a visitor pushes the switch the signal is sent to Raspberry Pi GPIO pins. When a Raspberry Pi gets signal it activates camera connected to it and takes a picture in JPG format. Capturing of signal by Raspberry Pi GPIO pins is implemented using the python script as shown in Figure 2.

```
File  Edit  Tabs  Help
pi@raspberrypi:~/miracl $ cat camguru.py
import RPi.GPIO as GPIO
import time
import subprocess
GPIO.setmode(GPIO.BCM)
GPIO.setup(18,GPIO.IN,pull_up_down=GPIO.PUD_UP)
print " "2019 S. Rajashree Research Scholar, Jain University "
print "Bhuvan Kumar P,  K S School of Engineering and Management, Bangalore"

while True:
    input_state = GPIO.input(18)
    if input_state == False:
        subprocess.call("./camguru",shell=True)
        time.sleep(0.2)
pi@raspberrypi:~/miracl $
```

Fig. 2. Python Script

```
File  Edit  Tabs  Help
pi@raspberrypi:~/miracl $ cat camguru

echo " "2019 S. Rajashree Research Scholar, Jain University "
echo "Bhuvan Kumar P,  K S School of Engineering and Management, Bangalore"
raspistill -o image1.jpg
convert -pointsize 100 -fill yellow -draw "text 50,110 '`date`'" image1.jpg imag
e.jpg
./ecsgen
./ecsign
zip cam_image.zip image.jpg image.ecs public.ecs
cp ./cam_image.zip /home/pi/backup/`date +"%Y-%m-%d-%H-%M-%S"`.zip
echo "copied zip file to /home/pi/backup/"
ls /home/pi/backup/
mpack -s "Test" /home/pi/miracl/cam_image.zip srajashrees@gmail.com
pi@raspberrypi:~/miracl $
```

Fig.3. Shell script

Step 2: After the image is taken by camera the image is fed as input to convert utility which is available in imagemagick library package that appends date and time when the image was taken. This is done by using shell script as shown in figure 3.

Step3: For authenticating the image this system uses ECDSA algorithm. The first phase of ECDSA algorithm is generation of keys (Private and Public keys). This system uses MIRACL crypto library for generating keys. This library has a C program file (ecsgen.c) that uses ECS curve parameter of p192 provided by NIST which is stored in common.ecs file. The ecsgen.c file internally calls mrcurve.c file that defines functions for point arithmetic. The ecsgen.c file takes seed (random number of 9 digits) as input to produce Keys and are stored in two files namely *private.ecs* and *public.ecs*.

The home security validation system uses current time as seed instead of random number to generate public and private keys as shown in Figure 4, Figure 5 and Figure 6. Public key is sent along with the original image file to the owner through mail. Private Key not sent in mail as it is kept confidential.

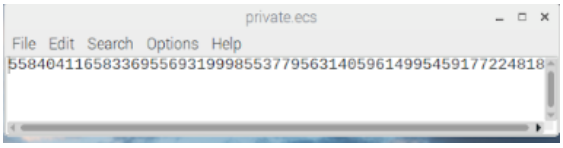

Fig.4. Private Key generated by ecsgen.cpp

Fig.5. Public key for the image sent by device

```
long seed;

 struct timeval t;

 gettimeofday(&t, NULL);

 seed = t.tv_sec * 1000000 + t.tv_usec;

 irand(seed);
```

Fig.6. Seed given to generate private key

Step 4: The second phase in the ECDSA algorithm is the signing of data. After getting the private key from ecsgen.c program it is used for signing of the image. Signature has two parts (R, S) as described in ECDSA algorithm. The MIRACL library has a program ecsign.c that takes input a random number and hash of the image file for generating signatures (equation 2 and 3).

Home security system proposed in this paper take current time as seed for ecsign.c program. The ecsign.c program creates and writes signature into *.ecs file by the same name as image file (*image.ecs)* which has two entries of 20 bytes for R and S as shown in Figure 7.

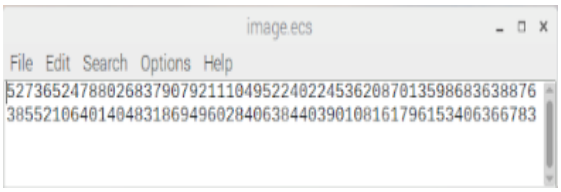

Fig.7. Signature of Image file sent by device

Step 5: After generating keys (public and private keys) and signatures (R, S value) the original Image along with public key and its signature are zipped using zip utility.

Step 6: The zipped file is stored in Raspberry Pi by creating current date and time as filename in backup folder.

Step 7: The zipped file is sent to the authorized owner mail using mpack utility as shown in Figure 8.

Step 8: The third phase in ECDSA is the verification phase that is achieved by the ecsver.c file of MIRACL crypto library that uses the public key (Q_A) stored in public.ecs along with the signatures (R, S) stored in image.ecs file. The verification process takes the image and finds the hash value of the image internally again and compares it with the value of R. For a valid image both should match if there is mismatch then the image is discarded.

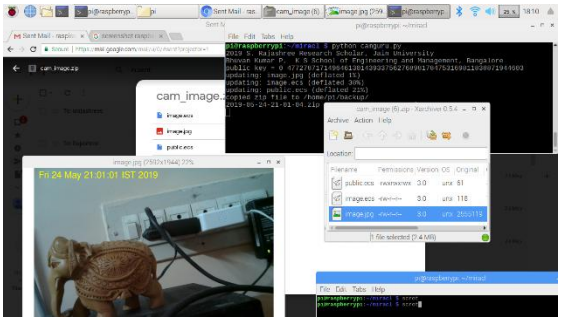

Fig.8. Screenshot of mail sent by device

The flow chart indicating the sequence of steps involved from pressing of tactile switch to sending of data to the monitoring e-mail address is shown below in Figure 9.

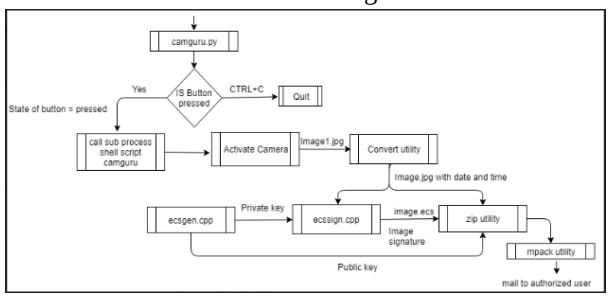

Fig.9. Process flow diagram.

V. RESULTS

Figure 10 shows the working of Python script which takes data from tactile switch, performs processing of image using MIRACL library and generates zip which is sent to registered e-mail address.

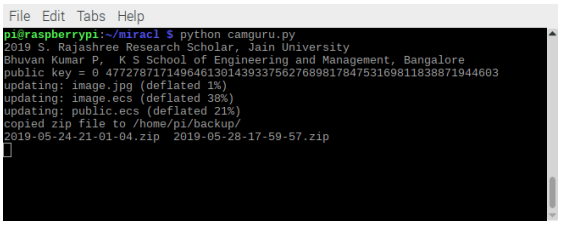

Fig.10. Working of python script

On receiving e-mail (containing zip file) the file is downloaded and unzipped and stored in appropriate folder. The verification program ecsver.cpp which is running at frontend (Windows or Linux) takes this file as input and verifies the digital signature and displays the image as shown in Figure 11 for Windows OS and Figure 12 for Linux OS. To display the image in Window Operating system *opencv[12]* package with visual Studio 2019 is installed.

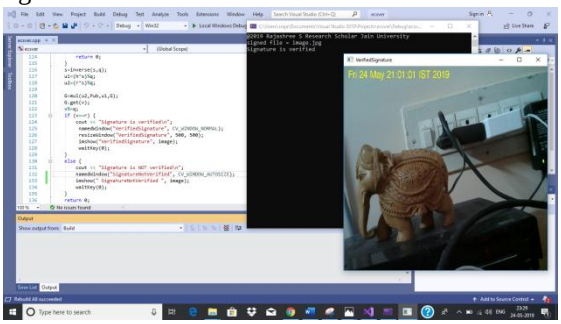

Fig.11. Screenshot of verification on Windows OS.

To display image in Linux operating system imagemagick package display utility is called from system () function.

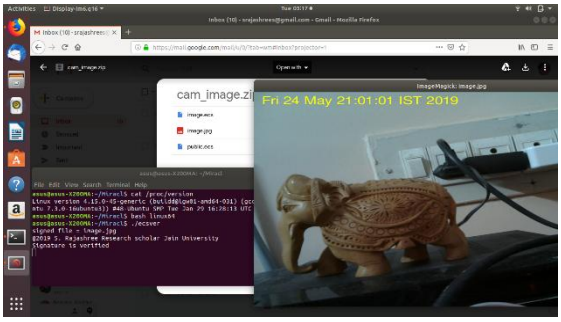

Fig.12. Screenshot of verification on Linux OS.

VI. CONCLUSION

Security mechanism using data capture and verification by ECDSA provides the advantage of high reliability in IoT environment and can be implemented at affordable cost. The proposed method uses the MIRACL crypto library to achieve this desired goal. Therefore, this paper proposes the enhancement to the existing mechanisms using ECDSA algorithm. The ECDSA generates digital signature that provides high level of data integrity by generating a private and public key. The data that is to send is signed by the private key and the signatures are verified by public key at the receiving side.

REFERENCES

1. Doshi, p.C.a.N., Internet of Things Security Challenges, Advances and Analytics 2019.

2. Rajashree, S., K.S. Soman, and P.G. Shah. Security with IP Address Assignment and Spoofing for Smart IOT Devices. in 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI). 2018.
3. Jia, X., et al., An efficient provably-secure certificateless signature scheme for Internet-of-Things deployment. Ad Hoc Networks, 2018. **71**: p. 78-87.
4. R. Rani, S.L., B. Poojitha, IoT Based Home Security System Using Raspberry Pi with Email and Voice Alert. International Journal of Advanced Research in Computer Science and Software Engineering. **8**(4).
5. Anvekar, R.G. and R.M. Banakar. IoT application development: Home security system. in 2017 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR). 2017.
6. Hussein, N.A. and I.A. Mansoori. Smart Door System for Home Security Using Raspberry pi3. in 2017 International Conference on Computer and Applications (ICCA). 2017.
7. 2019; Available from: https://thepihut.com/blogs/raspberry-pi-tutorials/16021420-how-to-install-use-the-raspberry-pi-camera.
8. ; Available from: https://github.com/miracl/MIRACL.
9. Williams, M.; Available from: http://ozzmaker.com/send-email-from-the-raspberry-pi-or-linux-command-line-with-attachments/.
10. [cited 2019; Available from: https://github.com/miracl/MIRACL.
11. 2018; Available from: https://raspberrypihq.com/use-a-push-button-with-raspberry-pi-gpio/.
12. opencv. opencv 2.4.9:[Available from: https://opencv.org/.

AUTHORS PROFILE

Rajashree Soman received the Bachelor of Engineering degree from Visvesvaraya Technological University, Belagavi, India, in 2003. She is pursuing Ph.D. degree from the Jain University, Bangalore, India, from 2017; currently she is working as Assistant Professor in PES University Bangalore, India. Her current research interests include Internet of Things, Cryptography, and IP security. Rajashree has coauthored 4 papers in peer reviewed conferences.

R. Sukumar completed his B.E. in ECE from Madurai Kamaraj University in 1992, M.E. in CSE from ManonmaniumSundaranar University in 2004, and Ph.D. from Anna University in 2010. He has over 20 years of teaching experience from undergraduate to Ph. D level. He is also a recognized Ph. D supervisor of Anna University Chennai and Jain University Bangalore. Already, three candidates have completed Ph.D under his supervision and six scholars are pursuing Ph.D. He is also serving as member of Board of Studies and various committees. He has over 25 high impact factor international journal publications and 20 international conference publications. He is currently serving as Professor at School of Engineering & Technology, Jain University, JGI Global Campus, Bangalore, India. His research areas of interest include Cryptography & Network Security, Sensor Networks, Cloud and IoT.

Bhuvan Kumar Panduranga: he is studying in Bachelor of Engineering degree in the department of Electrical and Electronics Engineering at K S School of Engineering and Management, Bangalore from Visvesvaraya Technological University, Belagavi, India. Domain areas of research are Internet of Things, Android application Development.