Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 165 (2019) 435–441

Procedia Computer Science

www.elsevier.com/locate/procedia

INTERNATIONAL CONFERENCE ON RECENT TRENDS IN ADVANCED COMPUTING 2019, ICRTAC 2019

Smart Metro Rail Ticketing System

Abhishek Nair M, Smit Taunk, Panyam Gangadhar Reddy, Parveen Sultana H*

Vellore Institute of Technology, Tamil Nadu, India

Abstract

Transportation plays a vital role in ones life. The main goal of this paper is to eradicate the day to day and one of the major problems with regard to carrying a ticket during transportation from ones life and make traveling a lot more peaceful. For this purpose, we are proposing a biometric-based ticketing system in the metro railway scenario but not limited to the same. In order to get a unique identifier for each person, we are considering their fingerprint right away from registration, booking tickets and validating the fingerprint on the day of the journey so he/she can travel on a particular day and on the desired train to his/her preferred destination. The fingerprint sensor will be interfaced with Arduino which in turn will store the fingerprint data to the cloud. We are proposing a two-way encryption standard for storing the sensitive fingerprint data in the cloud. This two-way encryption standard involves encrypting the data during data generation at the hardware end and encrypting it again before storing it in the cloud database.

Keywords: Biometric; Cloud encryption; Metro ticketing system; Security; Transportation.

1. Introduction

In the past, for any formal job of government, we required various different cards for user authentication, but with time all those cards were replaced by a single point of authentication Aadhar card in India. This card uniquely identifies an individual living in India. The main problem with the physical existence of a card is that we may sometimes forget to carry the card and due to carelessness, we may lose it and that can be further used by others for some different works. In such scenarios, the entire process of the ticketing system may become more complex to use. For travel, we need to carry our ticket as well as a UID card which is always a hectic task for day to day travel. So hence we propose a solution for a system that may take into account the biometrics of a person for authentication, which may be more useful than carrying a physical card around. So once a user registers himself/herself into the system using

* Corresponding author. Tel.: +91-944-310-6307
E-mail address: hparveensultana@vit.ac.in

10.1016/j.procs.2020.01.003

their fingerprint, they can travel from anywhere and any station across India, as they will not have to register again in this new station.

Now in order to save the fingerprint of the registered user and to be available anywhere, we will be using the cloud for storing the fingerprint data. Selectively, we will be using the Amazon AWSs DynamoDB for storing the fingerprint data received from the fingerprint scanner. DynamoDB is a NoSQL database that has very high distribution properties, by scaling up the database while storing a very large amount of data. Since the entire database is structure-less, we may not be worried about the fixed schema of the database too. These were some of the advantages of DynamoDB on the whole, but some of the common advantages of storing the data in the cloud are high availability, easy scalability, pay-as-you-go basis and so on.

Now since the sensitive fingerprint data of the user is going to be stored in the cloud, security becomes a major concern. With more and more registrations, the sensitive information is more prone to attacks from the third party and hence encryption of the data becomes more important. Here to overcome such problems, we will be using a Two-Way Encryption scheme which means that the data will be encrypted in two levels, one at the lower hardware level from the same place where the fingerprint data is being generated and one before storing the data into DynamoDB. We use the advanced Encryption Standard (AES) algorithm for encryption. It is a symmetric block cipher chosen by the U.S government to protect classified information and is implemented in software and hardware throughout the world to encrypt the sensitive data. It was a successor to the famous Data Encryption Standard (DES) which was starting to look vulnerable to brute-force attacks.

In this paper, we are providing a strategy to use the fingerprint data to ease the journey of the passengers in the Metro Station and completely avoid any paper-works. The rest of the paper is organized as follows. Next, it defines a survey of the works done in the area of fingerprint data encryption. Then it defines the existing work, against which we will compare our proposed model. Afterward, it will describe the proposed model with its algorithm and the Experimental setup. Finally, the paper ends with the Conclusion and Future Works.

2. Literature Review

The authors of [1] proposed a system where cloud can be used for storage and retrieving of the data in the IoT environment. They also mentioned that when the IoT system used in a real-time system then the cloud will be very much efficient in terms of availability. Cloud gives the easy integration, quick and easy to process complex data, with an added advantage of low installation cost. In our proposed project, we used the cloud for remotely accessing the data from anywhere, and also for easy processing. Coming to the data transfer from and to the cloud, the authors of [2] face two types of threats in cloud and IoT environments. One is data at rest and another is data in transit. Data in rest means when the data is in the cloud sitting idle. And data in transit means when data is in transmission from one node to another. To provide the security in both the aspects we used the two different algorithms for securing the data. Firstly we provide encryption in the Arduino itself using AES Algorithm so now the encrypted data goes to the cloud, where we process the data and encrypt again using the DES algorithm. So the data that is encrypted and secured both while sending and resting in the cloud.

IoT making our life easy but as its use is increasing making us vulnerable also. Authors of [3] suggest that the IoT is the future of our information technology as well as for electronic technology, as it deals with data, sometimes the data are very sensitive in terms of privacy and security at the time of sending can help to encrypt the data. We are using AES to encrypt data to store in cloud .we are using AES library in Arduino for encrypting the data of the fingerprint. The authors of [4] proposed a model for the hybrid encryption model for securing the diagnostic text data of medical images. As IoT gives us to enhance any system in big scale health sector is also adopting the IoT based system, but the health system is data is fully sensitive because most of the data is personal data so. Due to security reason before sending to the cloud the data should be encrypted.

For a Room Temperature Control and Fire Alarm/Suppression IoT Service using MQTT on the AWS system the authors of [5] uses the publish subscriber architecture and MQTT protocol, they used MQTT broker for connecting the IoT, Arduino is used to sending the data using Wi-Fi. Our main aim is to connect the Arduino (with Node MCU) to the AWS IoT Core in order to enable communication between them so that the devices can be controlled remotely. In Registration Device, we are going to get the fingerprint data of the new user which will be converted into a hexadecimal format. This data must be sent to the AWS IoT directly from Arduino using NodeMCU. The AWS IoT will have a

trigger that gets activated when it receives a message and then this fingerprint data is stored in the backend cloud database. While in the entry/exit devices on the journey day, we have to depict whether the user is allowed to travel or not, which will be shown with the help of LED. If a Green LED glows we allow the user and if the red LED glows when the user is not allowed to travel. Here the messages will be published by the Arduino device under the topic LED while the AWS IoT will serve as the subscriber to the LED topic. The state of the LED will be changed from the cloud end itself.

The authors of [6] have proposed the intricacies of the AES algorithm and a new method has been proposed to improve the security of the algorithm by introducing the dynamic S-Box Generation and Dynamic key generation. In this approach, more complexity is added to the data to increase Confusion and diffusion in the ciphertext by using the Dynamic Key Generation and then by using this S-box to make it difficult for the attacker to do any study of the S-Box. Authors of [7][8] have done a study of a new securing mechanism for low-power devices known as ECC (Elliptic Curve Cryptography). ECC is a public key encryption technique based on the elliptic curve theory that can be used to make faster and more effectual cryptographic keys. The keys are generated from the elliptic curve equation rather than the very traditional process of the generation as the product of very large prime numbers. Such a method of security is highly suitable for Arduino powered devices and also mobile devices as they provide a higher level of security compared to the RSA or DES with lower-key bits. This new method could be used for a future version of the same proposed system to make it more efficient and secure.

3. Existing Ticketing System

Currently, in the metro railway stations, they are using tickets or tokens for traveling. To travel, the customer needs to stand in the queue, need to buy a ticket for the destination, they need to carry the ticket along with them until they reach the destination. Fig.1 describes the existing ticketing system. If in middle they miss the ticket they will be penalized by the authority. The process carried out is they use the token system in which they use the RFID tag to give unique to each member[9]. While entering into the train, the customer needs to scan his token, the system will read the RFID tag and validates it and allows him to enter. The token is validated until the destination he cannot drop the train in middle or prolong his travel from his actual destination. After reaching the destination, the customer needs to scan the token with the system, if it is a valid journey the gates will open otherwise, the gates will not open he will be penalized.

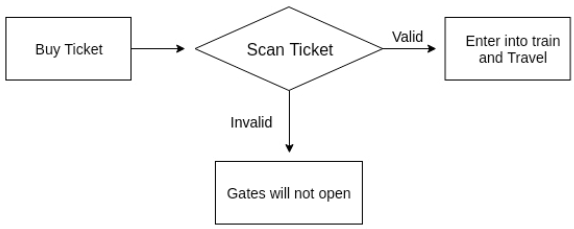

Fig. 1. Existing System

The data will be stored in the cloud they may be a chance of loss of data in the middle. The security system they are using can be breakable by hackers. The data is sensitive so we need to provide the security to the data. They provide the ticket scanners in which RFID will be present it will scan the tag and send the data to the cloud. The tag along with the source and destination will be sent to the cloud. When the person in the destination gives again his ticket it will scan for the tag and validates his tag in the cloud. To the validation, they use the searching algorithm and they take time to validate. The RFID tag that will convert into a string of data it should match all the literals present in the string then it will give results[10][11]. The condition needs to maintain is the number literals are the same as given if there is any difference in the literals it will be rejected so mainly depends on the algorithm implemented.

4. The Proposed System

The problems that travelers face in the metro station are, they can lose the ticket. The ticket may be stolen during the time of the journey. In that situation, they may be getting wrong results that will cause problems. While traveling if the person is missing his destination the RFID tag will not accept him as a valid customer in that situation he will be penalized accordingly. Some extra money needs to pay by the customer. The system needs to be fast as they will be a number of people the algorithm used need to be fast and should give the response. The ticket that was carrying a need to carry safely in case they lose that he cannot cross checkout gates. To overcome all these problems we are developing a system where there will be no need for a physical ticket and their fingerprints act as the ticket.

The problems that travelers face in the metro station are, they can lose the ticket[12][13]. The ticket may be stolen during the time of the journey. In that situation, they may be getting wrong results that will cause problems. While traveling if the person is missing their destination the RFID tag will not accept him as a valid customer in that situation they will be penalized accordingly. Some extra money needs to be paid by the traveler. The system needs to be fast as they will be a number of people the algorithm used need to be fast and should give the response. The traveler ticket need to carry the ticket safely in case they lose that he cannot cross checkout gates. To overcome all these problems we are developing a system where there will be no need for a physical ticket and their fingerprints act as the ticket.

Now coming to the data storage over the cloud, we first perform one more encryption algorithm on the already encrypted fingerprint data for more security and save it into the database in the cloud. Once this registration is done, the user moves to the book tickets portal. Here the traveler searches for trains according to the wish and book tickets using the registered fingerprint which they punched during the time of registration. After this, on the date of the journey, the user comes to ticketing kiosk and punches their fingerprint on the fingerprint sensor. If the data regarding the same person is present in the database with reference to the same date, the gates for that person will open and they can travel and at the end of journey at destination station, they again need to punch fingerprint so that they can come out of the station.

5. Architecture

Firstly, the fingerprint sensor will read the fingerprint image[14]. The sensor will store the data in the format of a 256-bit hexadecimal value. In the Arduino, the encryption process will take place by using the AES algorithm. The hexadecimal message is obtained from the fingerprint sensor that will be encrypted and send to the cloud. After sending the data to the cloud again encryption will take place by using another hybrid encryption algorithm, that the data will be encrypted the second time and will be placed in the cloud database. Fig. 2 demonstrates the proposed architecture .On the day who is getting to be boarded, they will be listed above in the table the data that is present in the cloud is ciphertext. The data that is present in the cloud is well secured as the data is in the two-way encryption format. In the cloud, the obtained data will be searched with cloud database content with the searching algorithm. If it founds the correct data it sends an acknowledgment to the Arduino. It will check the input and gives the response to the endpoint as valid or invalid.

Registration Workflow

- The user for the first time gives his/her fingerprint in the registration kiosk. The fingerprint data is recorded in the fingerprint module in the form of a 256-byte hexadecimal format.
- The fingerprint module is then interfaced with Arduino microcontroller, which will get the hexadecimal data from the fingerprint module and apply the AES algorithm on this collected data.
- The encrypted fingerprint data is then sent to the back-end python program from Arduino.
- The user enters his/her details such as the name, phone no. and date-of-birth into the registration webpage.
- The data entered is then sent to the Amazon AWS EC2 instance running in the cloud along with the AES encrypted fingerprint data.
- The data received at the EC2 instance is then parsed and the fingerprint data is sent to our hybrid encryption procedure which involves a combination of RSA and S-DES algorithm.
- The fingerprint data along with other details is then sent into the DynamoDB table in Amazon AWS that stores all the users' details who have registered.

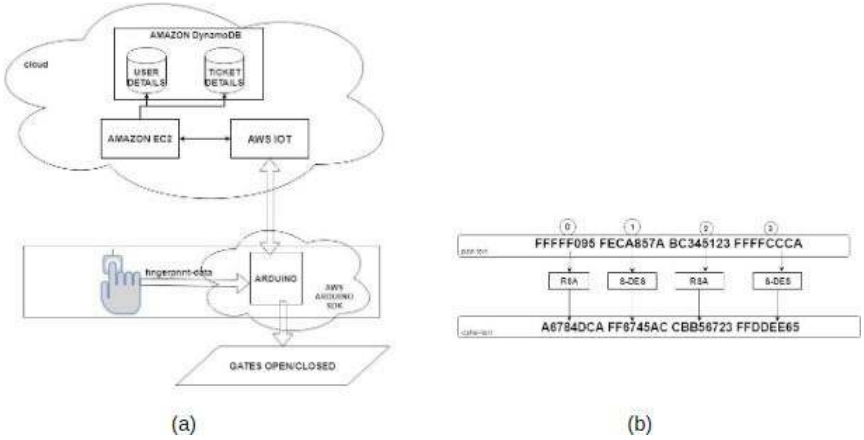

Fig. 2. (a)Proposed Architecture (b) Hybrid Encryption Algorithm

Architecture

Algorithm 1: AES Algorithm
Input: 256 bit-sized hexadecimal fingerprint data
Output: Encrypted 256 bit-sized ciphertext

1. Add Round key to plaintext with key k0.
2. Start the round.

   (a) Divide the plaintext as sub-bytes.
   (b)Perform the left shift of the plaintext.
   (c)Perform the Mix Column operation.
   (d)Add round key with key k1.

3.Continue the same process for 16 rounds.

Algorithm 2: Hybrid Encryption Algorithm
Input: 256 bit-sized hexadecimal fingerprint data
Output: Encrypted 256 bit-sized ciphertext

1.Partition the 256-bit data into 32 parts with 8 bits each.
2.Number each of the parts from 0 to 15.
3.Pass the odd-numbered partition into the RSA algorithm and record the cipher-text.
4.Pass the even-numbered partition into the S-DES algorithm and record the cipher-text.
5.Combine the cipher-texts from all the partitions and form another long 256-bit hexadecimal string.

Ticket Booking Workflow

- The user books the ticket by entering the details of the source, destination, and the date-of-journey.
- Then the user just enters his/her fingerprint is encrypted with AES and the data generated is then sent to the AWS EC2 instance.
- In this instance, we encrypt it using our encryption procedure and our searching algorithm finds the user's details in the DynamoDB table storing the registered users' data.

- If the user is a registered user then the page will provide an acknowledgment about the booked ticket and store the ticket-details into a new DynamoDB table.
- If the user is not registered, then he/she may have to go through the registration workflow again.
- Since the DynamoDB table is a NoSQL database, the entire table of stored tickets is partitioned based on the date-of-journey.
- On the day of the journey, the details of only those users who are going to travel on that particular day are going to be brought into the local ticketing kiosk.
- On the day of the journey, the user enters his/her fingerprint, and if the data matches the data stored in the ticketing kiosk, then the gates will open and the user will pursue his/her travel.

6. Results

We deployed the application on the localhost using Django, where the sites were primarily designed using HTML, CSS, and Python in the backend.

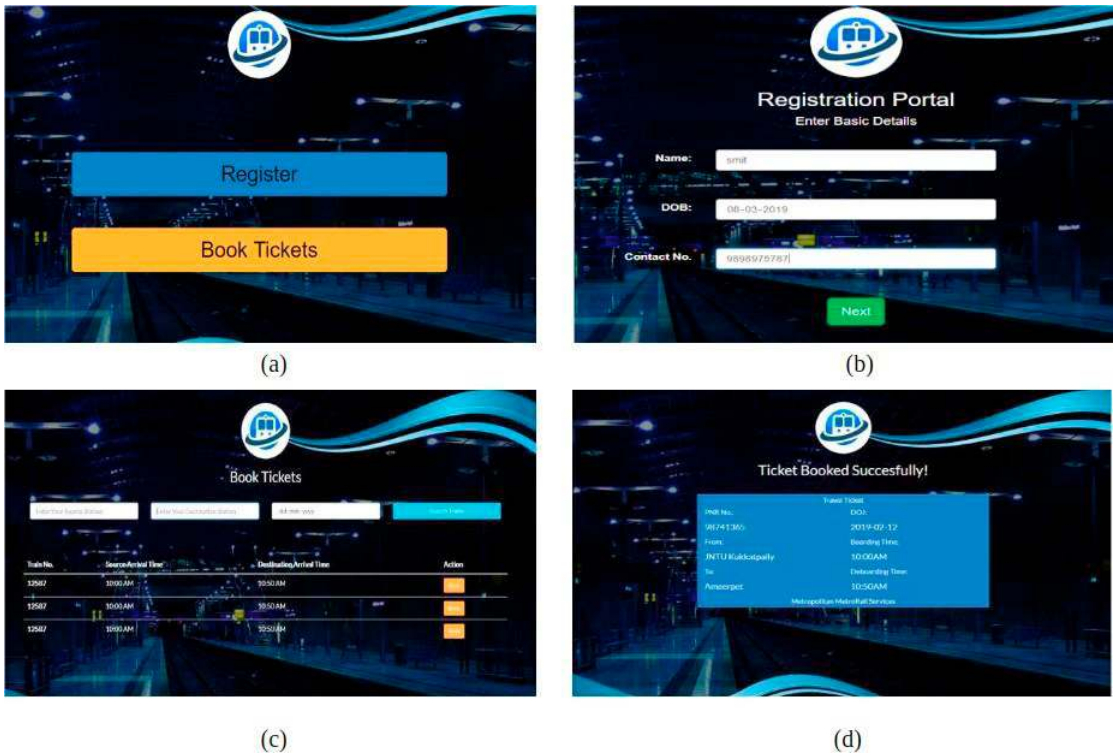

Fig. 3. (a) Webpage start screen (b) User registration screen (c) Ticket-booking page (d) Successfully generated Ticket

The very first screen of the web site where the registered user can either book the tickets for travel or a new user can register for the very first time as in figure 3. (a).

Once a new user enters the site, he/she is greeted with this page prompting the user to enter their basic details and these details being entered on this page are sent to the python backend to be sent to the Amazon cloud.

A registered user can book the tickets from this page where they have to enter the details about the journey (ie: source, destination and the date of journey) as in figure 3. (b). The details will be sent to the cloud which will then return a set of trains for the user to select from on that date of journey. The user selects any one of the trains from the list and enters his/her fingerprint as shown in figure 3. (c).

The final stage of the web application, giving an acknowledgment to the user about the booked ticket which will be then saved by the user as shown in figure 3. (d).

7. Conclusion

Using this system we eradicated the hassles of day to day traveling. There is now no need to carry physical tickets/tokens or any other UID card/documents for the sake of traveling. With this proposed methodology, the user will be ensured a more comfortable and convenient travel experience.

8. Future Works

A major future scope for this project is the use of ECC (Elliptic curve Cryptography) for securing the sensitive fingerprint data. Since ECC methodology is perfect for securing the data generated by low power devices such as Arduino. And since ECC provides the same level of security as that of RSA and DES, we can completely ignore the two-way encryption and provide only a single-level of encryption with ECC. Another prospect of this project is to implement the same application in the form of a mobile application, through which the user can book tickets directly from their mobiles and there is no need for a separate kiosk for that purpose.

9. References

[1] Atlam, Hany & Alenezi, Ahmed & Alshdadi, Abdulrahman&Walters, Robert & Wills, Gary. (2017). Integration of Cloud Computing with Internet of Things: Challenges and Open Issues. 10.1109/iThings- GreenCom-CPSCom-SmartData.2017.105.

[2] Albugmi, Ahmed &Alassafi, Madini& Walters, Robert & Wills, Gary. (2016). Data Security in Cloud Computing.

[3]S. M. Babu, A. J. Lakshmi and B. T. Rao, ”A study on cloud based Internet of Things: CloudIoT,” 2015 Global Conference on Communication Technologies (GCCT), Thuckalay, 2015, pp. 60-65.

[4]M. Elhoseny, G. Ramrez-Gonzlez, O. M. Abu-Elnasr, S. A. Shawkat, A. N and A. Farouk, ”Secure Medical Data Transmission Model for IoT-Based Healthcare Systems,” in IEEE Access, vol. 6, pp. 20596-20608, 2018.

[5]D. Kang et al., ”Room Temperature Control and Fire Alarm /Suppression IoT Service Using MQTT on AWS,” 2017 International Conference on Platform Technology and Service (PlatCon), Busan, 2017, pp. 1-5.

[6]F. J. D’souza and D. Panchal, ”Advanced encryption standard (AES) security enhancement using hybrid approach,” 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, 2017, pp. 647-652.

[7]A. Bansal and A. Agrawal, ”Providing security, integrity and authentication using ECC algorithm in cloud storage,” 2017 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, 2017, pp. 1-5.

[8]M. J. Kaur and P. Maheshwari, ”Building smart cities applications using IoT and cloud-based architectures,” 2016 International Conference on Industrial Informatics and Computer Systems (CIICS), Sharjah, 2016, pp. 1-5.

[9]Rijmen, V., &Daemen, J. (2001). Advanced encryption standard. Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology, 19-22.

[10]S. P. Jena, S. Aman, R. Das Computerized Green House Data Acquisition System Using Arduino with LabVIEW, International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 4, Issue 4, April 2015

[11]S. V. Devika, S. Khamuruddeen, S. Khamurunnisa, J. Thota and K. Shaik, Arduino Based Automatic Plant Watering System,International Journal of Advanced Research in Computer Science and Software Engineering 4(10), pp. 449-456, Oct. 2014

[12]D. M. Faris and M. B. Mahmood, Data Acquisition of Greenhouse Using Arduino, Journal of Babylon Univesity / Pure and Applied Sciences, No. 7, Vol. 22, 2014

[13]A. Bseiso, B. Abele, S. Ferguson, P. Lusch, and K. Mehta, A decision support tool for greenhouse farmers in low-resource settings, in Global Humanitarian Technology Conference (GHTC), 2015 IEEE, Oct 2015, pp. 292297.

[14]P. Ganguly, ”Selecting the right IoT cloud platform,” 2016 International Conference on Internet of Things and Applications (IOTA), Pune, 2016, pp. 316-320.