ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 6, June 2021

Secure Cryptographic Algorithm using Rijndael and Twofish with Timestamp Methodology in Remote Keyless Entry Systems

[1] Kunal Karnik, [2]Ajinkya Medhekar, [3]Manandeep, [4]Prof S.S Sambare
[1][2][3][4] Department of Computer Engineering, Pimpri Chinchwad College of Engineering, Pune

Abstract: In today's cyber era, user's privacy and safety have become the topmost priority for the automobile industry. As the world is transitioning towards the field of IoT, all the vehicles can communicate with the embedded devices within them such as the GPS, sensors, and cameras. Along with it, the interconnectivity between these vehicles is also evolving day by day. Keeping this in mind, we have proposed certain improvements by using the existing symmetric cryptographic algorithms to prevent cyber-attacks. Our technique mainly focuses on improving the locking and unlocking mechanism of the Remote Keyless Entry (RKE) using time-efficient and secure techniques.

Keywords—Vehicular security, Remote Keyless Entry (RKE) system, Twofish Cipher, Rijndael Cipher, Relay Attack, RollJam Attack

INTRODUCTION

The increase in the production of automobiles plays a significant role in enhancing the world economy. As modern vehicles rely on motherboards rather than axle grease, it has become evident for manufacturers across the globe to concentrate on the user's safety and vehicular security. Since most automobiles are getting connected to the internet via various means such as Bluetooth, GPS, and Wi-Fi, there has been a massive rise in the number of cyber-attacks performed on vehicles. Hackers are getting smarter with time and can exploit and hack the automobiles of top-class companies such as Volkswagen, BMW, Tesla, etc. Numerous flaws are present with the RKE systems of vehicles in terms of locking and unlocking mechanism. The amount of automobile cybersecurity incidents has increased to 605% since 2016. Out of these, 30% of the attacks were dealing with security breaches in the RKE systems. The wireless networking technology within automobiles has become possible because of the Radio-Frequency Identification (RFID) service. RFID binds devices with a unique serial number encoded within a tag. Radiofrequency between device tag and RFID reader enables the communication for identification and tracking of the given object. RFID devices can also support cryptographic functionality along with authentication. Modern cars have complex electronic circuits for providing access to drive and start it [6].

Attackers have always found a new to way penetrate the system by finding zero-day vulnerabilities in the system. Each locking mechanism will have a different kind of security, thus will be exposed to different kind of vulnerabilities. RKE systems were vulnerable to attacks like jamming attack, replay attack, rolljam attack [4], [6], [9], [13]. Many researchers found solutions to provide a security mechanism for RKE [1], [2], [3], [4], [7], [8], [13]. The solutions developed were complex and some of the solutions were not attack-proof like Hitag2 cipher. The solutions had many limitations (Tobias Glocker et. al, 2017) which used a 4-way handshake methodology for sending the data [2]. Furthermore, many solutions proposed used cryptographic schemes but used weaker ciphers.

Our contributions. We have developed a secure algorithm that can be implemented in the RKE system by automobile manufactures. Our algorithm provides a defense mechanism against a replay attack and rolljam attack. Our solution uses a unique combination of two symmetric cryptographic algorithms with a timestamp [18], [19], [21]. These cryptographic algorithms can rarely be found in the RKE system unlike other solutions using known ciphers like Playfair, Advanced Encryption Standard (AES). The solution is fast, secure, utilizes low computation power and easily implementable. The solution can be applied to the keys fobs and car lock by using a simple pseudo-random circuit for synchronization of random values and a real-time clock for timestamp methodology. We have described how we can implement the solution and scale it to a real-life scenario. We have added a detailed explanation about the procedure of attacks. Finally, an analysis is made of the proposed algorithm on different systems and results are compared in context to security and time.

Organization of the paper. The paper is organized as follows. Section II describes the background work related to the RKE system and different cryptographic schemes. Section III presents the proposed approach and the proposed model for RKE. Section IV presents an explanation of attacks and security analysis. Finally, the observations and result analysis are discussed in Section V, followed by the conclusion in Section VI.

BACKGROUND WORK

Zeinab et al. [6] have discussed detailed information about attacks and their prevention related to the communication of devices. They have introduced a three-layer framework by which the threats to automotive security can be understood. Leferink et al. [14] described the vulnerability against Pulsed Electromagnetic Interference They proposed an improved receiver design that consists of synchronous receivers, which are not very sensitive to pulsed interference. Verdult et al. [9] have presented several vulnerabilities in the Hitag2 transponders. They have executed the attacks on more than 20 vehicles of various make and model. Flavio Garcia and David Oswald [8] have demonstrated a cryptanalysis attack against Hitag2 stream cipher. Kobus Marneweck [13] explained the KEELOQ cipher, which is a block cipher based on 32-bit length. Vulnerabilities in the KEELOQ cipher were found by Bogdanov [11] in February 2007. This attack is based on the slide technique and a linear approximation of the non-linear Boolean function. Further. Sebastiaan Indesteege [12] extended the attack by combining a slide attack [11] with a novel meet-in-the-middle approach to recover the key from a slid pair. Tobis Glocker et al. [2] had designed a Lightweight Symmetric Encryption Algorithm using an Electrical Erasable Programmable Read-Only Memory (EEPROM) and a challenge-response technique. Jinta Patel [1] found security weaknesses in Glocker et al. [2] proposed model, which allowed the attacker to access the car without the actual key fob. Jinita Patel [1] proposed a protocol that is secure against OBD port scan attack including other attacks such as relay, replay and impersonation. Another approach proposed by Selvakumar et. Al [3] was the RKE Crypto model designed with two methodologies: double encryption in the last 32-bit data and single encryption in the last 64-bit data. The RKE Crypto Model uses Playfair cipher [17] for the encryption and decryption process. Kyle Greene [23] used a timestamp mechanism with AES to prevent replay attack in RKE. Vanesa Daza and Xavier Salleras [4] proposed LASER (Lightweight And Secure Remote keyless entry) protocol which mitigates attacks using Blake2 (hash function) for RKE and PRKE. Deepali Rane [21] explained the superiority of Twofish over the Blowfish algorithm and experimental results proved TwoFish requires lesser time than BlowFish. According to the analysis made by IEEE for AES vs Twofish, it was observed that the AES algorithm is faster for text encryption but with a sufficient increase in RAM, the Twofish algorithm was faster [18]. The size of the RAM affects the Twofish algorithm [16], [18], [21]. Niansheng Liu [22] made a comparison between RC6 and Rijndael and stated RC6 needs more CPU and memory resources than Rijndael. In 1997, NIST announced the open challenge for a block cipher [15] and Rijndael was the winner of the competition. It had fewer round and provided good security with low computation power [26].

We have observed most of the cryptographic algorithm used are symmetric because they fast and use less memory [1], [2], [3], [23]. Timestamp was used by Vanesa [4] in LASER and Kyle Greene [2] for preventing a replay attack. The next section describes the proposed algorithm for RKE system security.

PROPOSED WORK

Our proposed work solely focuses on the Remote Keyless Entry (RKE) system security. In our proposed work we have tried to replicate a real-life scenario of locking or unlocking a car. Normally the owner of the car will press a button on the key fob and the car will respond accordingly. We have used two Raspberry Pi to represent a car and a key fob. We have used the Raspberry Pi 3B+ model. The transmission and receiving frequency used is 433 MHz, so we have used the 433 MHz RF modules. The RF modules have few drawbacks like Distance (problem solved using copper coil), Noise and Smaller Memory. The other components used are our jumper wires, push button buzzer, LED lights, power supply. As the push button is pressed, the process of locking/unlocking will begin. The lights and buzzer will get activated on the completion of the process. The architecture of the proposed system can be understood from Fig 1. We have explained the proposed work in two-part: first, we have explained the transmission and receiving process within the Raspberry Pi. Secondly, in the proposed algorithm section, we have described the procedure for encryption and decryption of data.

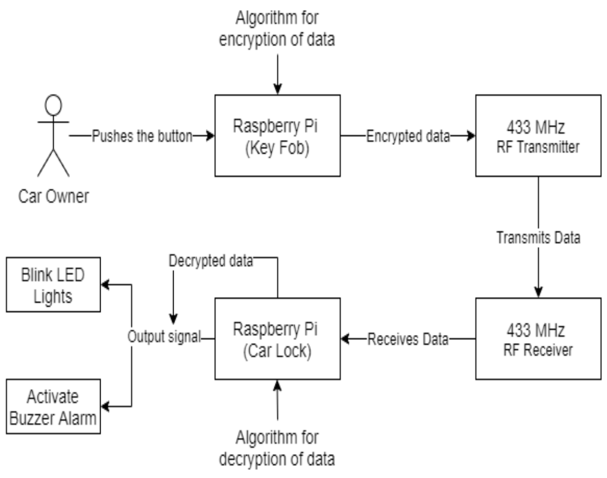

Fig. 1 System architecture of the proposed model

3.1 Proposed Approach

1. To demonstrate the process of locking and unlocking the car using a key fob, we have used two Raspberry Pi (Model 3B+) and 433 MHz RF module.
2. The first Raspberry Pi will act as the key fob for transmitting the radio frequency. A push-button and RF transmitter are attached to the first Raspberry Pi via the breadboard. As the button is pressed, it will initiate the process of transmission.
3. The encryption algorithm takes place in the first Raspberry Pi and then transmits these encrypted bits over the frequency.
4. The second Raspberry Pi will act as the car lock. The transmitter will send the data to the second Raspberry Pi and the RF receiver will receive the transmitted frequency. We have attached a buzzer, LED light and RF receiver to the second Raspberry Pi via the breadboard.
5. Once the bits are received, the data is decrypted in the second Raspberry Pi. After decryption, the data is checked with certain conditions like whether the integrity of the data is maintained or not and accordingly the algorithm will proceed.
6. If we consider, there was no malicious activity performed during this process, an output signal is sent. This output signal will flash the LED lights and activate the buzzer alarm to depict a scenario of locking and unlocking the car.

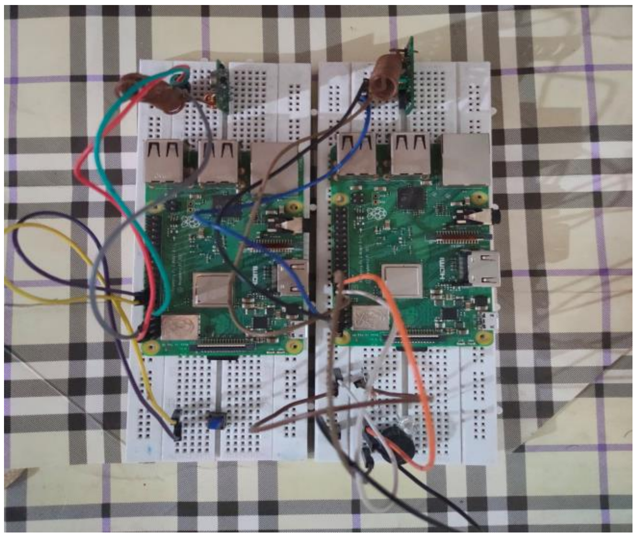

Fig. 2 Hardware of the proposed model

3.2 Proposed Algorithm

The proposed algorithm focuses on enhancing security using symmetric cryptographic algorithms. We have used Rijndael and Twofish cryptographic algorithms. We have created a unique combination using both algorithms to provide higher security. These algorithms used are very powerful and provides an alternative to the Advanced Encryption Standard (AES) algorithm [15], [16], [17], [18], [19], [20], [21], [22].

Rijndael is a symmetric key encryption algorithm that is constructed as a block cipher. The algorithm has a total of 10, 12, or 14 rounds and with key sizes of 128 bits, 192 bits, or 256 bits, respectively. Rijndael uses a combination of three discrete and invertible layers or uniform matrix transformations: Linear Mix Transform, Non-linear Transform and Key Addition Transform. Rijndael was the winner of a competition ran by the National Institute of Standards and Technology (NIST) for an industry standard for encryption in the year 2000/2001. [15], [22], [26]

Twofish is a symmetric key encryption algorithm that is constructed as a block cipher. It has a block size of 128 bits, with keys 128 bits, 196 bits and 256 bits. It is based on a Feistel network and has a total of 16 rounds. The TwoFish algorithm is an alternative to the Blowfish algorithm and requires lesser time than BlowFish [21]. Twofish uses pre-computed, key-dependent S-boxes, which means that the S-box is already provided but is dependent on the cipher key to decrypt the information. Twofish features a very complex key schedule The AES algorithm is fast but with a sufficient increase in RAM, the

Twofish algorithm was faster for text encryption.[16], [18], [19], [20]

3.2.1 Encryption process on 256-bit data

Algorithm 1 shows the encryption process. Message m is our input data of size 128-bit. Twofish cipher is called for the first part of the encryption process. For the first encryption process, we have defined both the first key (k1) and block size (blocksize) as 128-bit data. Twofish requires a total of 3 parameters: Message (m), Key (k) and Block size (blocksize). We store the output of Twofish encryption in e1. Now, e1 is added with the current timestamp of size 128-bit indicated by start_ts. This addition of data is stored in a temporary variable called tmp. Rijndael cipher is called for the second part of the encryption process. For the second encryption process, we have defined the second key (k2) as 128-bit and block size (blocksize) as 256-bit data. Rijndael requires the same parameters as Twofish but instead of message m, we will pass the tmp variable. We store the output of Rijndael encryption in e2. The 256-bit e2 data is transmitted to the receiver along with the command cmd. The command tells whether the car should lock or unlock.

Algorithm 1: Encryption of data

```
Input: m ← 128-bit data
Output: e2 ← 256-bit encrypted data
  1: k1, blocksize ← 128-bit, 128-bit
  2: e1 ← encrypt_Twofish (m, k1, blocksize)
  3: start_ts ← 128-bit current timestamp
  5: tmp ← e1 + start_ts
  6: k2, blocksize ← 128-bit, 256-bit
  7: e2 ← encrypt_Rijndael (tmp, k2, blocksize)
  8: return e2
```

3.2.2 Decryption process on 256-bit data

Algorithm 2 shows the decryption process of 256-bit data received. The process will be exactly the opposite of the encryption process. First, we call the Rijndael cipher for decryption. The output will be stored in d2. The first 128-bit data is stored in d1 and the last 128-bit data is stored in start_ts. On d1 we apply the Twofish cipher and get the original message m. For authentication, we store the difference between 128-bit end timestamp (end_ts) and 128-bit start timestamp (start_ts) in a variable called diff_ts. Message m, difference of timestamp diff_ts and command cmd are sent to the authentication algorithm.

Algorithm 2: Decryption of data

```
Input: e2 ← 256-bit encrypted data
Output: m ← 128-bit decrypted data
        diff_ts ← 128-bit data [end_ts – start_ts]
  1: k2, blocksize ← 128-bit, 256-bit
  2: d2 ← decrypt_Rijndael (e2, k2, blocksize)
  3: d1 ← First 128-bit of d2
  4: start_ts ← Second 128-bit of d2
  5: k1, blocksize ← 128-bit, 128-bit
  6: m ← decrypt_Twofish (d1, k1, blocksize)
  7: end_ts ← 128-bit current timestamp
  8: diff_ts ← end_ts – start_ts
  9: return m, diff_ts
```

3.2.3 Authentication for decrypted 256-bit data

Algorithm 3 shows the authentication process. In this process, we need to check for two conditions c1 and c2. The first condition states that if the original message m is equal to the decrypted message. This verification is done to check whether any bits were altered or not. The second condition states the difference of timestamp diff_ts should be less than or equal to a given threshold value, say X. This condition will authenticate if there was any attempt of an attack on the car because the attacker may jam the car and send the signal. Thus, causing an increase in the difference of timestamp and making the value greater than the threshold value. If these two conditions are satisfied then the car is locked or unlocked, else the car will not respond to the signal.

Algorithm 3: Authentication of decrypted data

```
Input: m ← 128-bit data
       diff_ts ←128-bit data
       p ← 128-bit data
       t ← X ms
       cmd ← Lock or Unlock.
Output: Accept ← Locking/Unlocking of car
Reject ← No response from car
1: c1 ← isEqualTo (m, p)
2: c2 ← isLessThanEqualTo (diff_ts, t)
3: if (c1 AND c2) then
4:       Accept: execute cmd
5: else
6:       Reject: no execution of cmd
7: endif
```

The following figure 3 shows the working sequence of all

ISSN (Online) 2394-2320

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 6, June 2021

the three algorithms: Encryption of data, Decryption of data, Authentication of decrypted data.

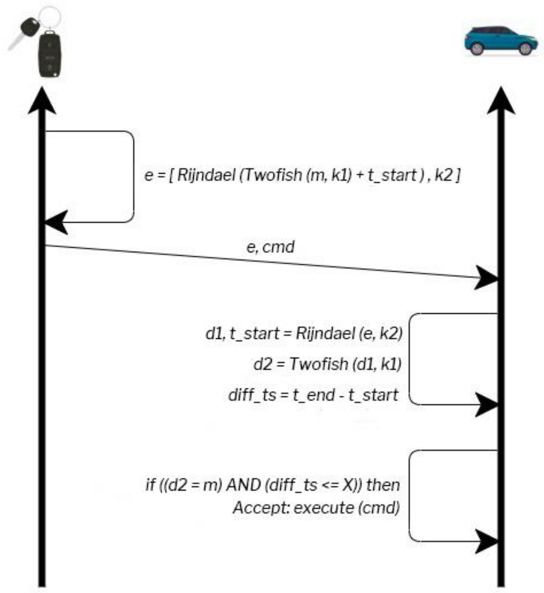
e = [Rijndael (Twofish (m, k1) + t_start) , k2]
e, cmd
d1, t_start = Rijndael (e, k2)
d2 = Twofish (d1, k1)
diff_ts = t_end - t_start
if ((d2 = m) AND (diff_ts <= X)) then
Accept: execute (cmd)

Fig. 3 Proposed Model for RKE

The following flowchart summarises the working of the proposed approach.

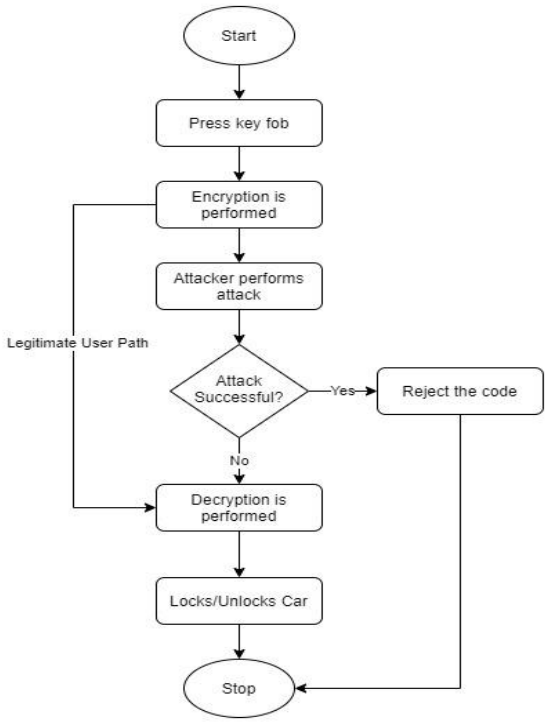

Fig. 4 Proposed Approach Flowchart

ANALYSIS OF PROPOSED MODEL

4.1 Attacks

4.1.1 Replay Attack:

A replay attack falls under the category of network attack. In this attack, the attacker detects a data transmission and unethically has it delayed or repeated. This helps the attacker to gain access to a network, gain information or complete a duplicate transaction. A replay attack is also termed a playback attack. Figure 5 shows how a replay attack is performed.

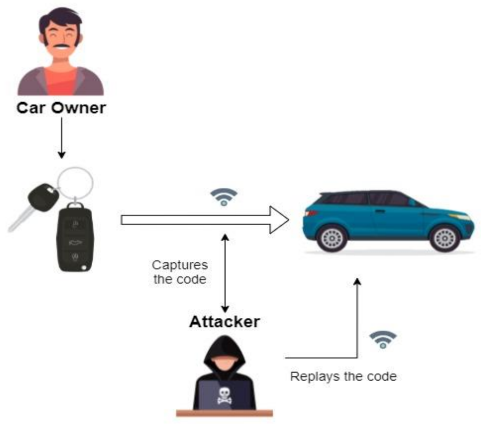

Fig. 5 Replay Attack

4.1.2 RollJam Wireless Attack

The RollJam attack involves rolling codes replaced fixed codes as so a security measure. The attacker first jams and stores signal sent from the key fob and fail to lock or unlock the car. This forces the user to press the button again. On the second press, the attacker again jams the signal and stores the second code and transmits the first code, locking or unlocking the car. The second code is still unused which can be later used. Figure 6 shows how a rolljam attack is performed.

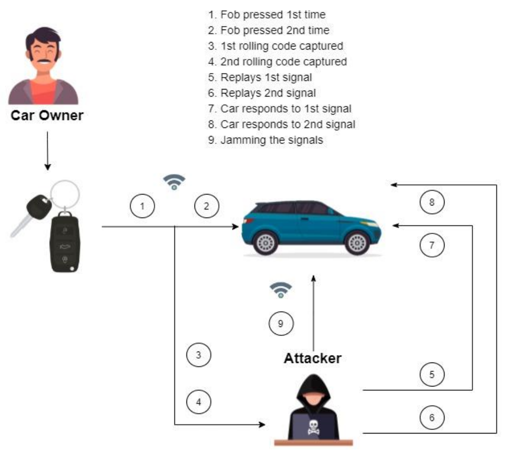

Fig 6. RollJam Attack

4.2 Communication Modes

To understand the process of an attack, we need to understand the transmission modes available. The data is sent from one point to another and can be received back to the source. We have a total of 3 Transmission Modes: Simplex, Half-Duplex and Full-Duplex.

1. Simplex: In simplex mode, we have unidirectional communication. It means the data can be sent from one source but cannot be received back. Example: NooElec NESDR Smart Software Defined Radio. This hardware can be used to receive the transmitted frequency. It can be used with free software like SDR#, SDR-Radio, Linrad, GQRX.
2. Half-Duplex: In half-duplex mode, both the nodes can transmit and receive the data. This process cannot take place at the same time, which means the node cannot be in transmission as well as in receiving state at the same given time. Example: YARD Stick One. YARD Stick One comes with RfCat firmware allowing us to control the YARD Stick One from an interactive Python shell.
3. Full-Duplex: In full-duplex mode, both the nodes can transmit and receive the data parallelly or simultaneously. It can be used when communicating in both directions, thus the node is transmitting and receiving at the same time. Example: LimeSDR (Software Defined Radio). The LimeSDR works as a transmitter and receiver. LimeSDR is an open-source Software Defined Radio.

4.3 Implementation of Attacks

4.3.1 Replay attack

A replay attack has a simple procedure. We need to capture the code and replay it back. For this type of attack, we require a half-duplex communication mode. We can use a YARD Stick One or NooElec NESDR Smart to capture the code. Once we get the code the transmitter (YARD Stick One) will replay the code.

4.3.2 RollJam attack

RollJam attack can be performed using a $32 radio device founded by Samy Kamkar [24] or a full-duplex device like LimeSDR [4]. Alternative suggestions can be to use two YARD Stick One, the first one will be used for jamming the signal and the second one for receiving the signal. This process can be easily automated as YARD Stick One firmware supports python. Another alternative would be to use a YARD Stick One and NooElec NESDR Smart. In this scenario, YARD Stick One will jam the signal and we can receive the signal using NooElec NESDR Smart. The signal received is recorded to a .wav file and can be opened in an audio editor software like audacity. The signal needs to be manually demodulated and then transmitted using a YARD Stick One.

Note: The hardware may have other alternatives. We can use a different transmitter or receiver for performing the attacks.

4.4 Security Analysis

4.4.1 Replay attack

To prevent replay attack, our solution sends a unique encrypted value of plaintext. Each time a unique encrypted string is generated and is accepted by the receiver. Hence, an attacker will not be able to replay this message again because we are using rolling secret keys.

4.4.2 RollJam attack

To prevent a rolljam attack, our solution ensures the encrypted message sent by the fob will not be useful to the attacker. Our solution verifies the difference of timestamp during authentication. The attacker stores the message to replay it later and therefore, due to this delay in replaying the t_start value will generate a larger difference in the timestamp. This will guarantee to exceed the threshold value as the difference between timestamps is larger. Additionally, the timestamp is also encrypted which makes the attack more difficult.

To prevent the above attacks, we have ensured that the secret keys are not exposed to a brute-force attack. The secret keys are randomised using uppercase and lowercase letters, digits, and special symbols. We have used two different keys for encryption and decryption which makes it difficult to brute-force. To brute-force the one secret key we will require the following number of combinations:

Number of combinations required = 2 ^ 128

In this equation, we have used the power of 2 because the key is in binary format. The key is 128 bits; therefore, the total numbers of combinations will be 2 raised to the power of 128. Thus, we can evaluate the above equation.

OBSERVATION AND RESULT ANALYSIS

The proposed model is implemented using basic hardware to test. The hardware used in a real-life scenario is different. Observation during experimentation:

- 433 MHz modules cannot process a large amount of data. The data should be sent in chunks.
- The Raspberry Pi should be used with an alternating current rather than a power bank for better performance.
- We have used a fixed seed value to randomise the data, but in an actual scenario, the pseudo-random sequence generator circuit will generate a random secret key.
- The randomisation should be such that the key fob and car lock's secret key should synchronize.

The following graph in figure 7 shows an analysis of computation time (in ms) for the proposed model. We have tested the algorithm on 4 different machines.

1. Machine 1: Broadcom BCM2837B0 A53 (ARMv8), 1 GB RAM, 8 GB Memory
2. Machine 2: Intel i3 7th Gen, 12GB RAM, 1 TB HDD
3. Machine 3: Intel i5 8th Gen, 8GB Ram, 1 TB HDD
4. Machine 4: Intel i7 8th Gen, 8GB RAM, 512 SSD

From fig 7, we can see the proposed model takes less than a second to compute. The RKE Crypto [3] computation time is more as to our proposed model. Our proposed model provides more bits than the RKE Crypto model. This increases the security in our proposed model and additionally, provides an appropriate amount of time for the RKE system to function.

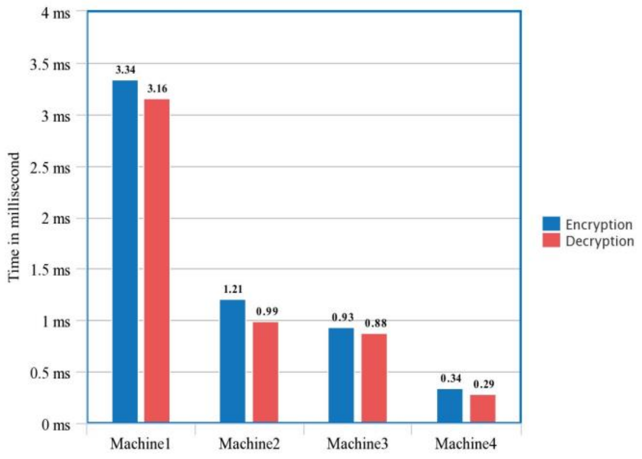

Fig 7. Computation time for encryption and decryption

CONCLUSION

In this paper, we have proposed a new model which uses a unique combination of Rijndael and Twofish symmetric cryptographic algorithms with a timestamp mechanism to increase security. The proposed model is implemented using Raspberry Pi and 433 MHz RF modules. Our solution is better in terms of security and computation time as Rijndael and Twofish provide a higher level of security and it takes less time for computation. The solution provides a defense mechanism against the attacks described in section IV and using two rolling secret keys with timestamp makes it even more secure. The computation time is fast and takes less than a second for the entire process to complete as we can observe from the graph (Figure 7). The hardware we used had the lowest specification among other machines, but it still could produce the results in an average time of 3.4ms and 3.1ms for encryption and decryption, respectively. Lastly, the proposed model can be easily implemented within the RKE systems and proves its effectiveness for the RKE system.

REFERENCES

[1] Jinita Patel and Manik Lal Das, "On the Security of Remote Key Less Entry for Vehicles", IEEE International Conference on Advanced Network and Telecommunication Systems (ANTS), 2018.

[2] Tobias Glocker and Timo Mantere, "A Protocol for a Secure Remote Keyless Entry System Applicable in Vehicles using Symmetric-Key Cryptography", 8th International Conference on Information and Communication Systems (ICICS), 2017.

[3] Madhumitha Sri Selvakumar, Rohini Purushoth Kumar, Gunasekaran Raja, "Effective Cryptography Mechanism for Enhancing Security in Smart Key System", IEEE Tenth International Conference on Advanced Computing (ICoAC), 2018.

[4] Vanesa Daza and Xavier Salleras. “LASER: Lightweight and Secure Remote keyless entry protocol”, Arxiv, 2019.
[5] Prof. Sahu, SonaliLole, “Vehicle Theft Alert & Engine Lock System Using ARM7”, International Journal of Advance Research and Innovative Ideas in Education (IJARIIE), 2017.
[6] Zeinab El-Rewini, Karthikeyan Sadatsharan, “Cybersecurity challenges in vehicular communications”, Elsevier, 2019.
[7] Omar A. Dawood, Othman I. Hammadi, “An Analytical Study for Some Drawbacks and Weakness Points of the AES Cipher (Rijndael Algorithm)”, 1st International Conference on Information Technology (ICoIT’17), 2017.
[8] Flavio D. Garcia et al., “Lock It and Still Lose It – On the (In) Security of Automotive Remote Keyless Entry Systems”, 25th USENIX Security Symposium, 2016.
[9] Roel Verdult, Flavio D. Garcia, Josep Balasch, “Gone in 360 Seconds: Hijacking with Hitag2”, Conference: 21st USENIX Security Symposium, 2012.
[10] Roberto Di Pietro, Gabriele Oligeri, "Jamming mitigation in cognitive radio networks", IEEE Network, 2013.
[11] Andrey Bogdanov, "Linear Slide Attacks on the KeeLoq Block Cipher", Springer International Conference on Information Security and Cryptology, 2007.
[12] Sebastiaan Indesteeg, Nathan Keller, "A Practical Attack on KeeLoq", Citeseerx, 2008.
[13] Kobus Marneweck, “An Introduction to KEELOQ Code Hopping”, http://ww1.microchip.com/downloads/en/AppNotes/91002a.pdf
[14] Stefan van de Beek and Frank Leferink. “Vulnerability of Remote Keyless-Entry Systems against Pulsed Electromagnetic Interference and Possible Improvements”, In IEEE Transactions on Electromagnetic Compatibility, 2016.
[15] Andreas Dandalis, Viktor Prasanna, Jose Rolim, "A Comparative Study of Performance of AES Final Candidates Using FPGAs", Springer International Workshop on Cryptographic Hardware and Embedded Systems, 2002.
[16] S.A.M. Rizvi, Syed Zeeshan Hussain, Neeta Wadhwa, "Performance Analysis of AES and TwoFish Encryption Schemes", International Conference on Communication Systems and Network Technologies, 2011.
[17] Reena Singh, Shaurya Taneja, Kavneet Kaur, "Modified play-fair encryption method using quantum concept", International Conference on Computing for Sustainable Global Development (INDIACom), 2016.
[18] Cloud Storage Information, https://cloudstorageinfo.org/twofish-vs-aes-encryption
[19] Anil G. Sawant, Dr Vilas N. Nitnaware, “Twofish Algorithm for Encryption and Decryption”, Volume 6 Issue 1, Journal of Emerging Technologies and Innovative Research (JETIR), 2019.
[20] Aparna K, Jyothy Solomon, “A Study of Twofish Algorithm”, International Journal of Engineering development and research (IJEDR), 2016.
[21] Deepali Rane, “Superiority of Twofish over Blowfish”, International Journal of scientific research and management (IJSRM), 2016.
[22] Niansheng Liu, Jianjun Cai, "Cryptographic Performance for Rijndael and RC6 Block Ciphers", IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID), 2017.
[23] Kyle Greene, Deven Rodgers, “Timestamp-based Defense Mechanism Against Replay Attack in Remote Keyless Entry Systems”, IEEE International Conference on Consumer Electronics (ICCE), 2020.
[24] Samy Kamkar Blogs, http://samy.pl/
[25] Juan Wang and Karim Lounis, “CSKES: A Context-based Secure Keyless Entry System”, IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), 2019.
[26] Finjan Cybersecurity Blog, https://blog.finjan.com/rijndael-encryption-algorithm