On the Security of Remote Key Less Entry for Vehicles

Jinita Patel
DA-IICT
Gandhinagar, India
jinitapatel24@gmail.com

Manik Lal Das
DA-IICT
Gandhinagar, India
maniklal_das@daiict.ac.in

Sukumar Nandi
IIT Guwahati
Assam, India
sukumar@iitg.ernet.in

***Abstract*—With the rapid growth in the automobile industry and the advancement in the embedded technology, the traditional mechanical key for operating vehicles is gradually replaced by Remote Keyless Entry (RKE) system. Enabling RKE system in vehicles not only improves the security of car access but also facilitates convenience to users. At the same time, RKE system introduces many threats such as eavesdropping, relay, replay attack, On-board Diagnostic (OBD) port scan attack, key fob cloning, jamming and so on. In this paper, we discuss on the keyless car entry system, observe some security weaknesses in a recent RKE system and present a new RKE system. The proposed RKE system uses the notion of unclonable security module to minimize the OBD port scan attack including other known threats. The proposed RKE system achieves authentication of car as well as key fob by preserving privacy of the communicating entities. The security analysis and experimental results show that the proposed RKE system is secure and practical with respect to efficiency and user convenience.**

***Keywords*—Vehicle security; Remote Keyless Entry system; OBD Port Scan; Secret Unknown Cipher; Authentication.**

I. Introduction

Automobile industry is one of the fastest growing industries that has been contributing significantly to world's economy. While automobile sectors continually leverage societal growth, users convenience and security of vehicles become the prime concerns to car manufacturers. When mechanical locks are employed to lock/unlock the vehicles door as well as to operate the engine of the vehicles, the locks are easy to break or clone, and importantly, they are not very convenient to users in comparison to modern keyless vehicle operation system. In order to improve the security of vehicles, automobile industry is transforming continually new technologies into the vehicle system. The vehicle entry systems such as Immobilizers [1], Passive Keyless Entry (PKE) [2], [3], [4] and Remote Keyless Entry (RKE) system [5], [6], [7] are being used in modern vehicles. Out of these, the RKE system has found popularity with respect to car's security and user convenience. In RKE system, user carries a key fob that contains a short range radio transmitter and receiver. The key fob has buttons to operate the vehicle. The user has to press a button on the key fob to lock/unlock the vehicle. Once the vehicle authenticates key fob, the vehicle processes the requested instruction. Car manufacturers use different authentication techniques such as fixed code [8], rolling code [9] and challenge-response. Rolling code technique is widely used in the RKE system and garage door opening systems [9] [10]. Rolling code technique implemented using KeeLoq block cipher [5], which was found insecure [11], [12], [6] in which an attacker can recover secret key used in KeeLoq block cipher. Moradi et al. [13] proposed a RKE system that is resistant to power analysis attacks. Flavio et al. [14] show that an attacker can clone the original key fob's memory state and thereby, can get unauthorized access to the vehicle by recovering the secret keys from electronic control units of vehicles. They also show Hitag2 rolling code technique, which is used in many car manufacturers, is insecure by performing the correlation-based attack. Lv and Xu [10] proposed rolling code authentication technique based on Advance Encryption Standard (AES) encryption algorithm [15]. In order to prevent the vulnerabilities in rolling code technique, car manufacturers prefer to use challenge-response based authentication. Glocker et al. [7] discussed a challenge-response based authentication protocol for implementing RKE system in car. We show security weaknesses in [7] by which an attacker can extract the entire memory state of a key fob, which allows the attacker to access the car without the actual key fob.

Our contributions. In this paper, we first discuss about the security weaknesses that we found in Glocker et al.'s protocol [7]. We show how an attacker can reveal the secrets stored in key fob and car that are used to build the authentication message response in [7]. Once the attacker cloned the key fob, he can steal the vehicle using the cloned key fob. We present a new RKE system using the notion of unclonable function. In our proposed RKE system, the key fob stores a set of challenge-response pairs which is used for authenticating the car, when the key fob triggers any command to the car. The car's OBD stores only the set of challenges and the corresponding responses are being computed by car's SUC (secret unknown cipher) board as and when any command comes from the key fob. The challenge-response message between the car and the key fob is exchanged in an encrypted form and the car's identity is protected in our RKE system, which does not give any advantages to adversary while eavesdropping, OBD scanning or replaying message. The analysis and experimental results of the proposed protocol show that the proposed RKE system is secure, practical and convenient to users.

Organization of the paper. The remaining of the paper is organized as follows. Section II discusses the background and related works. Section III presents the proposed RKE system for car's security. Section IV provides the security strengths and experimental results of the proposed RKE system. We conclude the paper in Section V.

II. Background and Related Work

In a fixed code technique [8], one fixed code is initially saved in a vehicle and key fob. Upon pressing a button on the key fob, the key fob transmits its fixed saved code. Vehicle checks if the fixed code transmitted by the key fob is valid or not. If the code is correct, then the vehicle executes requested instruction. In a rolling code technique [9], [10], key fob transmits different code in each transmission. One sequence counter is maintained in the key fob and the vehicle. One shared secret is stored in key fob and vehicle. When user triggers a action, the content of the sequence counter is transmitted after encrypting it using the secret key. After each transmission, the value of the sequence counter is incremented. The car decrypts the received message sent by the key fob. The vehicle then check the key fob's sequence counter content with the content of its own sequence counter. If the difference between the two sequence counter's content is within a certain predefined range, then the vehicle verifies the message from the key fob and executes the requested instruction. Then the vehicle increments its own sequence counter after receiving a correct code. In the challenge-response technique, a secret key is shared between vehicle and key fob. The vehicle sends a random challenge to key fob when key fob requests for any service (e.g. lock/unlock). The key fob may perform keyed hash based authentication or standard encryption [15] using a shared secret key and sends the response to the vehicle. Upon receiving the response from the key fob, the car verifies response with its own computed message. If the received string and computed string matches, the requested instruction gets executed [8]. In a Passive Keyless Entry System (PKES), key fob is activated automatically and locks/unlocks the car when key fob is within the car range. The work [16] and [17] addresses the issue and solution for the correct estimation of proximity between car and key fob. Glocker et al. proposed a challenge-response based authentication protocol [7] for RKE system. In their protocol, key fob and car both stores the array of 2000 random numbers in a range between 0 to 65535. When user presses a button on the key fob, it sends car identity to the car. The car generates 10 random numbers and send it to the key fob after verifying car identity. Received 10 random numbers generated by car are used as an indices of the memory locations whose values are used to build authentication message. Key fob sends response to the car's challenge. After receiving the authentication message from key fob, car verifies by comparing received authentication message with the constructed authentication message. As OBD port scan attack is vulnerable in challenge-response mechanism, the attackers can unlock vehicles using cloned key fobs. A device called OBD key programmer is available in the market to clone key fobs. The attacker plugs a device into the OBD port of the vehicle which allows them to download the sufficient vehicle's electronic information to create a compatible key fob to steal the vehicle. This process can take just seconds [18]. To mitigate OBD port scan attack, the protocol in [7] informs driver of the car to update the key fobs. During the updating process, the board computer generates new random numbers and writes it to the memory of the key fobs and the vehicle. We explain how the attacker can recover the random numbers stored in the memory of car and key fob in [7], by which the attacker can obtain the entire memory state of key fob and OBD of the vehicle.

A. *Weaknesses in Glocker et al.'s protocol [7]*

In [7], authentication message is sent in public channel, which provides the sum of the contents of indices in plain text. We show that if an attacker finds three transitive pairs within same or different session then he can obtain the content of indices.

Assume that M[2000] is an array of 2000 random numbers and ten random numbers generated by car are $x_1, x_2, x_3, x_4, x_5, y_1, y_2, y_3, y_4, y_5$. The AM[] is an array of authentication message, where

$$AM[1] = M[x_1] + M[y_1]$$

$$AM[2] = M[x_2] + M[y_2]$$

$$AM[3] = M[x_3] + M[y_3]$$

$$AM[4] = M[x_4] + M[y_4]$$

$$AM[5] = M[x_5] + M[y_5]$$

When user presses a button on the key fob, key fob sends car identity to the car. The car generates 10 random numbers and sends it to the key fob. The attacker blocks the reception of these 10 random numbers and sends another 10 random numbers in pattern a, b, a, a, a, b, c, c, d, e to the key fob. Five pairs for encryption are (a, b), (b, c), (a, c), (a, d) and (a, e). The key fob generates authentication message response using these random numbers and sends it to the car. The car will not authenticate key fob as response is incorrect. Thus, the user will try again. Attacker record this communication and from this he can get values of $a + b$, $b + c$, $a + c$, $a + d$ and $a + e$. Suppose that s_1, s_2 and s_3 are the sum of contents of two memory locations. Now,

$$s_1 = M[a] + M[b] \tag{1}$$

$$s_2 = M[b] + M[c] \tag{2}$$

$$s_3 = M[a] + M[c] \tag{3}$$

$$s_1 - s_2 = M[a] - M[c] \tag{4}$$

From using Eq. (3) and Eq. (4) the attacker knows the content of indices a, b and c. After knowing this, the attacker knows the content of indices in which a, b and c are paired. Therefore, the attacker can obtain the content of indices d and e and eventually the entire memory state of the key fob and OBD of the car after a few more sessions.

We note that challenge-response based authentication works in many other real-world application including automobile industry. However, the design goal and adversarial assumption for such protocols need to be carefully thought of with respect to end-user device capability, intended security, practicality and user convenience. In next section, we present a new RKE system for vehicles security, which can resist the potential threats as mentioned in the above discussion.

III. Proposed RKE Protocol

A. RKE System Model

The system consists of two entities – User (Key fob) and Car (OBD). The key fob and the car use a common secret based on the shared index between them in a particular session. Both key fob and car's OBD are personalized by the car manufacturer in a secure manner. The secret keys are not stored in car, instead, they are generated when needed during the challenge-response protocol between the key fob and the car. For generating encryption keys, the proposed protocol employs secret unknown cipher [19] for key derivation and message encryption purpose.

B. Secret Unknown Cipher (SUC)

Unclonable unit such as Physical Unclonable Function (PUF) [22] has been used for authentication and key storage. A PUF uses physical diversity within each object and produces unique and unpredictable mapping to each response. The responses of PUF are unpredictable as it depends on operating conditions such as voltage, temperature and radiation. Aging effect changes the material properties with time, and thereby, mapping also changes with time that makes responses of PUF non-consistent. To overcome this inconsistency, digital PUFs are introduced [20]. Secret Unknown Cipher (SUC) [19] is a self-created internal permanent digital structure which can encrypt and decrypt, where aging effects are negligible, and hence, they are consistent in the whole lifetime of digital products. Indeed, PUFs are harder to clone in comparison to digital PUF. However, due to practical implementations digital PUFs are widely used [20]. Realizing such SUC unit requires embedding it in a self re-configuring non-volatile technology, preferably in System-on-Chip (SoC) Field Programmable Gate Arrays (FPGA) units [19], [21]. In SUC creation process, one unknown irreversible cipher from a vast library of theoretically infinite classes of ciphers is installed in SUC, which can process key derivation, encryption/decryption, authentication code and similar cryptographic operations.

C. Key fob and OBD personalization process

Key fob and car's OBD personalization is performed by the car manufacturer in a trusted environment. While programming key fob and OBD personalization, one SUC unit is embedded in car's OBD. The controller of SUC unit generates random numbers to create all the challenges C_i. The controller gives challenge C_i to the SUC_{car} and gets the corresponding response R_i after computation. This procedure is repeated for all the n challenges to get the corresponding n responses. After that, the controller sends back all the challenges C_i and responses R_i to the key fob. Key fob stores all the n challenges and corresponding n responses. The OBD of the car stores only challenges. We consider the length of the car identity is of 32 bits, the challenges of 32 bits and responses of 128 bits. Storage cost is dependent on number of challenge response pairs in proposed protocol. Fig. 1 shows the personalized Key fob and Car's OBD in the proposed RKE system.

The car identity is not communicated in plaintext, instead, a pseudo-identity is generated from the actual car identity in every session which is shared between the car and the key fob. During the key personalization process, the initial pseudo-identity is created on car's SUC using the parameters of car's actual identity and the secret key known to car's SUC. The pseudo-identity is then stored on both car's OBD and key fob. For every challenge-response session, the key fob sends the pseudo-identity to the car, which must be verified by the car in order to accept any instruction from the key fob. After every successful session, both the key fob and the car update the pseudo-identity at their storage as (new)pseudo-identity = PRF(challenge, response, (old)pseudo-identity), where PRF() is a cryptographically secure pseudo-random function, and challenge, response, (old)pseudo-identity are the parameters used in the current run of the protocol.

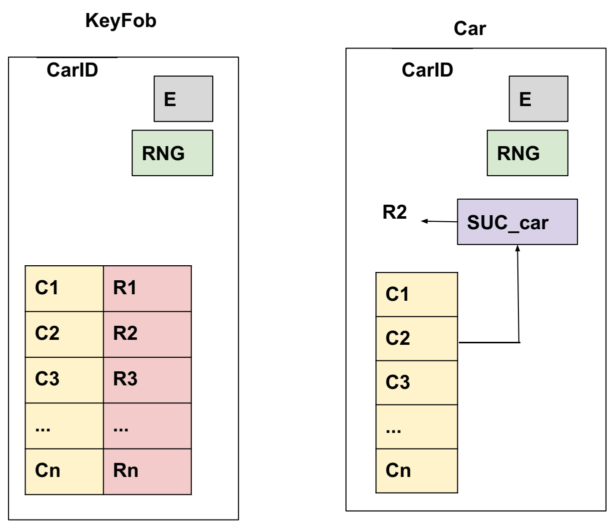

Fig. 1. Personalized Key fob and Car's OBD in the proposed RKE system

D. Challenge-Response between Key fob and Car

The proposed challenge-response protocol between key fob and car consists of `Send Command`, `Verify Command`, `Send Challenge`, `Verify Challenge`, `Send Response` and `Verify Response` algorithms, which are explained as follows. When button is pressed on a key fob, the `Send Command` function is invoked. The `Send Command` function is captured in Algorithm 1. Similarly, `Verify Command` is captured in Algorithm 2, `Send Challenge` is captured in Algorithm 3, `Verify Challenge` is captured in Algorithm 4, `Send Response` is captured in Algorithm 5 and `Verify Response` is in Algorithm 6. The work-flow diagram of the proposed challenge-response protocol is depicted in Figure 2.

Algorithm 1 Key fob sends command to the car

```
1: procedure SendCommand(pseudo_car-id, $r$)
2: if (Button is pressed) then
3:     Generate a random number $r$
4:     Send pseudo_car-id, $r$, instruction command to car
5: end if
```

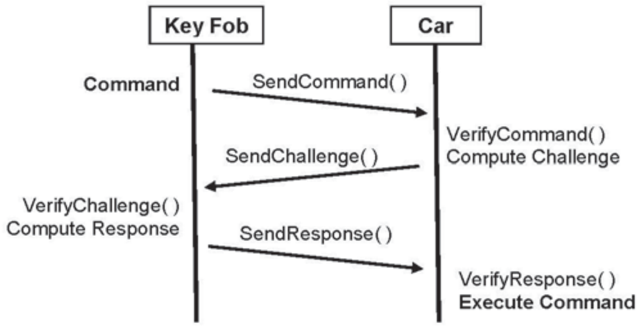

Fig. 2. The work-flow of the challenge-response protocol

Algorithm 2 Key fob's command verification

```
1: procedure VerifyCommand(SendCommand(pseudo_car-id, $r$))
2: if (SendCommand() is received) then
3:    if (Received pseudo_car-id = stored pseudo_car-id) then
4:       Invoke SendChallenge()
5:    end if
6: end if
```

Algorithm 3 Challenge computation and sending it to key fob

```
1: procedure SendChallenge($i \| E_{R_i}\{C_i \| (r+1) \| t\}$)
2: Selects randomly an index $i$ from $n$ available indices
3: Compute $R_i = SUC_{car}(C_i)$
4: Generate a random number $t$
5: Compute $E_{R_i}\{C_i \| (r+1) \| t\}$
6: Send $i \| E_{R_i}\{C_i \| (r+1) \| t\}$ to key fob
```

Algorithm 4 Challenge verification

```
1: procedure VerifyChallenge(SendChallenge())
2: if (SendChallenge($i \| E_{R_i}\{C_i \| (r+1) \| t\}$) is received) then
3:    Get the $R_i$ corresponding to its storage of index $i$
4:    Decrypt($E_{R_i}\{C_i \| (r+1) \| t\}$ using the key $R_i$
5:    if (Decrypted $r$ = SendCommand(., $r$,.) then
6:       Invoke SendResponse()
7:    end if
8: end if
```

Algorithm 5 Response computation and sending it to car

```
1: procedure SendResponse($E_{R_i}\{(r+2) \| t\}$)
2: Compute $E_{R_i}\{(r+2) \| t\}$
3: Send $E_{R_i}\{(r+2) \| t\}$ to car
```

Algorithm 6 Response verification and key fob's command execution

```
1: procedure VerifyResponse($E_{R_i}\{(r+2) \| t\}$)
2: Decrypt $E_{R_i}\{(r+2) \| t\}$ using the key $R_i$
3: if (Decrypted $t$ = SendChallenge(., $t$,.)) then
4:    Execute the instruction of SendCommand()
5: end if
```

We note that the length of the challenge is of 32-bit and for the corresponding response is 128-bit length. The attacker can gather these set of challenge-response pairs by intercepting the communication between the same car and the key fob. However, the challenge-response pair is captured in encrypted form. The challenge length is kept 32-bit for efficiency, which can lead to guessing it by brute force. To defend this threat, the proposed protocol wants the key fob be periodically re-personalized for updating a new set of challenge-response pairs in car's OBD and key fob's memory. During the re-personalization process, the new challenges and responses are generated with SUC based on the key personalization process and written them onto the memory of the key fob and the car's OBD. Although the re-personalization process adds some overhead periodically to the challenge-response protocol, the protocol resists the stated threats and preserves privacy of the car.

IV. Analysis of the Proposed Protocol

A. Security Analysis

A usual threat to a challenge-response protocol is, if an attacker can predict or gather previously intercepted challenge-response and then replay the same to impersonate to either challenger, responder or both. Therefore, the challenge-response mechanism should resist replay attack. The relay is another potential threat in remote keyless signal to vehicles. To defend relay attack, distance bounding approach works to a certain extent with an added cost on top of the main challenge-response exchanges. In addition, the attacker should not be able to distinguish which of the challenge-response pair links to what service and at what context. Car's OBD port scanning attack and key fob cloning are real challenges to automobile industry that demand tamper-resistant memory state of embedded devices which can resist these threats. We analyze the proposed challenge-response protocol and show its strengths against the above stated threats.

Claim 1. The proposed protocol preserves the privacy of car and user (key fob).

Proof. In the protocol, the real identity of the car is not communicated, instead, a pseudo-identity is generated for every session. During the Key personalization process, the initial pseudo-identity is created on car's SUC using the parameters Car's real identity and the secret key. Then, the pseudo-identity is stored on both car's OBD and key fob. For every challenge-response session, the key fob sends the pseudo-identity in encrypted form along with the random number. Upon receipt of the command from key fob, the car decrypts it and check whether the pseudo-identity stored on car's OBD matches. If so, it proceeds further with the protocol. At the end the successful protocol run, both the key fob and car update the pseudo-identity at their storage as (new)pseudo-identity = PRF(challenge, response, (old)pseudo-identity), where PRF() is pseudo-random function, and challenge, response, (old)pseudo-identity are the parameters used in the current run of the protocol. In the protocol, the key fob and the car authenticate each other based the secret parameters exchanged in the protocol session. The random numbers used in every session make the challenge-response indistinguishable from the previous run of the protocol, even though the same key fob and car participant in the protocol. Therefore, the protocol preserves the privacy of car and user (key fob). □

Claim 2. The proposed protocol defends OBD scan and response stealing attacks.

Proof. The OBD scan attack in proposed RKE protocol is difficult as attacker has to guess the dynamic response message of 128-bit at the correct time. Although, the attacker can guess the car-id which is a 32-bit string, the new challenge-response pairs are generated periodically with SUC based on the Key personalization process and written them onto the memory of the key fobs and the car's OBD. Furthermore, the attacker can not get access to secret keys/responses as they are not stored in OBD's memory. Instead, the keys are generated when they are needed with the car's SUC, which is tamper resistant physical identity module [19]. As a result, the proposed protocol resists OBD scan attacks. □

Claim 3. The proposed protocol resists replay attacks.

Proof. In the proposed protocol, suppose that an attacker tries to use the information intercepted from some previous session, that is, replaying the message to convince the car that the communication is a fresh command initiated by the key fob. If the replays the `SendCommand`() of a previous session, then the car detects it by seeing the old random random in its temporary storage. If the attacker constructs a fresh `SendCommand`() by using a new random number, then it will get detected in the `SendResponse`() phase in which the attcker will not be able to provide the correct response R_i (as it is kept secret at the key fob's memory) corresponding to the challenge C_i comes from the car. Therefore, the attacker will not be able to replay the previous message without being caught in the proposed protocol. □

B. Experimental Results

The proposed RKE system consists of key fob and car's OBD. Radio transceiver is employed in key fob as well as in the car. When user presses a button on the key fob, the transmitter generates radio signals in a free usable frequency band 315 MHz or 433 MHz. Fig. 3 shows implementation setup of the proposed RKE system.

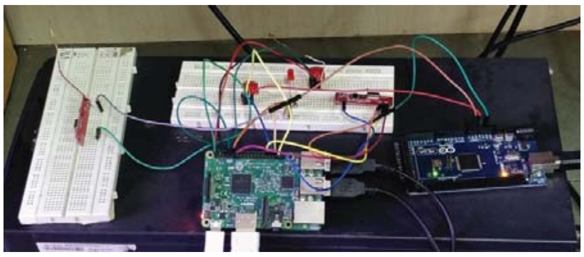

Fig. 3. Implementation setup of the proposed RKE system

For key fob and car's OBD we used Raspberry Pi 3 Model B and for RF transmitter and receiver we used ASK transmitter module ST-TX01-ASK and ASK receiver module ST-RX02-ASK, respectively, which operates on 433 MHz. We implemented the protocol using the piVirtualWire library of Python. We note that in real OBD's system SUC must be implemented in non-volatile memory and ciphers installed in SUC should be unknown. We simulated the SUC environment with SHA-512 with secret salt as a irreversible function of SUC to generate responses. Fig. 4 and Fig. 5 show computations time (in msec) in [7] and in the proposed RKE system, respectively.

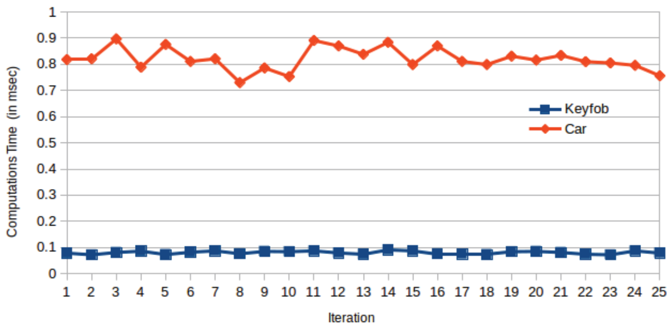

Fig. 4. Computations time (in msec) in Glocker et al.'s protocol [7]

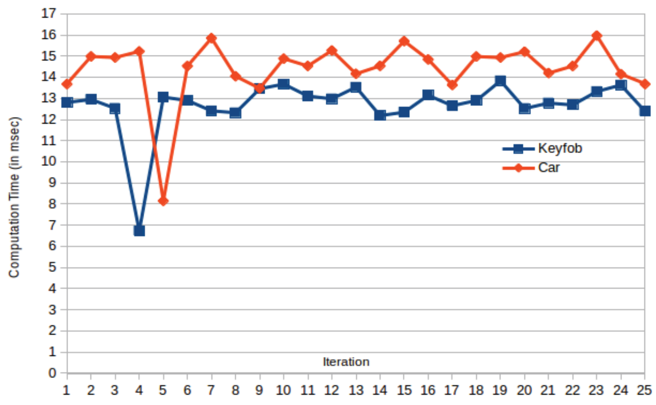

Fig. 5. Computations time (in msec) in the proposed PKE system

V. Conclusion

We have discussed a recent protocol [7] proposed by Glocker et al. for RKE system in vehicle security. We showed that the protocol [7] is not secure, suffers from OBD port scan attack, which is one of the important security features in RKE system in automobile industry. We have proposed a protocol for RKE system which is secure against OBD port scan attack including other attacks such as relay, replay and impersonation. The proposed RKE system is implemented in a testbed using Raspberry Pi 3 and piVirtualWire library. It is observed that the computational time of key fob in the proposed RKE system is 13 msec, which is acceptable in practical scenario where both key fob and car's OBD mutually authenticate each other and then execute the instruction intended in the protocol run.

Acknowledgment

This research was supported in part by the Indo-French Centre for the Promotion of Advanced Research (IFCPAR) and the Center Franco-Indien Pour La Promotion De La Recherche Advancée (CEFIPRA) through the project DST-CNRS 2015-03 under DST-INRIA-CNRS Targeted Programme.

References

[1] M. Knebelkamp and H. Meier. Latest generation technology for immobilizer systems. Texas Instruments, http://www.ti.com/tiris/docs/manuals/whtPapers/immobilizer.pdf

[2] J. Waraksa, K. Fraley, R. Kiefer, D. Douglas, and L. Gilbert. Passive keyless entry system. https://www.lens.org/images/patent/US/5319364/A/US_5319364_A.pdf.

[3] Microchip Technology Inc. Passive keyless entry (PKE). User's manual. http://ww1.microchip.com/downloads/en/DeviceDoc/PKERefDesignMnl.pdf

[4] Tiny Single Chip Passive Keyless Entry. https://www.nxp.com/products/identification-and-security/secure-car-access/

[5] Microchip Technology Inc. TB001: Secure Learning RKE Systems using KeeLoq Encoders. http://ww1.microchip.com/downloads/en/AppNotes/91000a.pdf

[6] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, and B. Preneel. A Practical Attack on KeeLoq. In Proceeding of the Theory and Applications of Cryptographic Techniques (EUROCRYPT), LNCS 4965, Springer-Verlag, pp. 1–18, 2008.

[7] T. Glocker, T. Mantere, and M. Elmusrati. A protocol for a secure remote keyless entry system applicable in vehicles using symmetric key cryptography. In Proceedings of International Conference of Information and Communication Systems, pp. 310-315, 2017.

[8] I. Alrabady and M. Mahmud. Analysis of attacks against the security of keyless-entry systems for vehicles and suggestions for improved designs. IEEE Transactions on Vehicular Technology, 54(1):41–50, 2005.

[9] K. Marneweck. An introduction to Keeloq code hopping. In Technical Report: Microchip Technology Inc, 1996.

[10] X. Lv and L. Xu. AES encryption algorithm keyless entry system. In Proceedings of International Conference on Consumer Electronics, Communications and Networks, pp. 3090–3093, 2012.

[11] T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, and M. Shalmani. On the power of power analysis in the real world: A complete break of the KeeLoq code hopping scheme. In Proceedings of International Cryptology Conference - Advances in Cryptology (CRYPTO), LNCS 5157, Springer, pp. 203-220, 2008.

[12] A. Bogdanov. Attacks on the KeeLoq block cipher and authentication systems. In Proceedings of the Conference on RFID Security, 2007.

[13] A. Moradi and T. Kasper. A new remote keyless entry system resistant to power analysis attacks. In Proceedings of International Conference of Information, Communications and Signal Processing, pp. 1-6, 2009.

[14] D. Garcia, D. Oswald, T. Kasper, and P. Pavlides. Lock it and still lose it - on the (in)security of automotive remote keyless entry systems. In USENIX Security Symposium, 2016.

[15] A. Menezes, P. van Oorschot, and S. Vanstone. Handbook of Applied Cryptography. In CRC Press, 1996.

[16] Z. Gong, A. Ozen, Y. Wu, X. Cao, R. Shin, D. Song, H. Jin, and X. Bao. Piano: Proximity-based user authentication on voice-powered internet-of-things devices. In Proceedings of International Conference on Distributed Computing Systems, pp. 2212–2219, 2017.

[17] J. Zhang, Z. Wang, Z. Yang, and Q. Zhang. Proximity based IoT device authentication. In Proceedings of Conference on Computer Communications, pp. 1–9, 2017.

[18] D. Rubino. How crooks can steal your car without the key. https://www.autocar.co.uk/car-news/industry/how-crooks-can-stealyour-car-without-key.

[19] E. Hamadaqa, A. Mars, W. Adi, and S. Mulhem. Clone-resistant vehicular RKE by deploying SUC. In Proceedings of International Conference on Emerging Security Technologies, pp. 221-225, 2017.

[20] M. Fyrbiak, C. Kison, M. Jeske, and W. Adi. Combined HW-SW adaptive clone-resistant functions as physical security anchors. In Proceedings of NASA/ESA Conference on Adaptive Hardware and Systems, pp. 130-137, 2013.

[21] W. Adi, A. Mars, and S. Mulhem. Generic identification protocols by deploying secret unknown ciphers. In Proceedings of International Conference on Consumer Electronics, pp. 255-256, 2017.

[22] C. Herder, M. Yu, F. Koushanfar, and S. Devadas. Physical Unclonable Functions and Applications: A Tutorial. In Proceedings of IEEE, 102(8):1126–1141, 2014.