# CPN Based Analysis of In-Vehicle Secure Communication Protocol

Rustam Rakhimov Igorevich[1,2(✉)], Daekyo Shin[1], and Dugki Min[1,2]

[1] SoC Platform Research Center, Korea Electronics Research Institute,
Daewangpangyo-ro 710beon-gil, Seongnam,
Gyeonggi-do 463-400, Republic of Korea
{rustam, dukeshin}@keti.kr,
{rustam, dkmin}@konkuk.ac.kr
[2] School of Computer Science and Engineering,
College of Information and Telecommunication,
120 Neungdong-ro, Gwangjin, Seoul 143-701, Republic of Korea

**Abstract.** Security in the domain of In-Vehicle communication becomes critical issue when modules from different vendors allowed interacting with car. Authentication and information secrecy issues must be solved by car vendors. Many research works were held about authentication of car accessory devices and their secure communication with central unit (HUD). In this work we have analyzed one of the recently published In-Vehicle Secure protocol. Multiple replay attacks were discovered during analysis of the protocol. CPN (Coloured Petri Nets) tool was applied to anlyze and demonstrate the flaw in given secure message exchange protocol.

**Keywords:** In-Vehicle · Secure · Protocol · HUD (Head Unit Display) · CPN (Colored Petri Nets) · Replay attack

## 1 Introduction

Nowadays modern cars contain dozens of controllers that are increasingly networked together via various bus communication systems. Basically those networks were connected to non-critical controllers of the car, such as: light control, window control, door locker etc. In modern cars networks have access to several life critical components of the vehicle, like breaks, airbags and engine control. Those modern cars that are equipped with driving aid systems like ESC (Electronic Stability Control) or ACC (Adaptive Cruise Control) allow deep intervention in the driving behavior of the vehicle. Third party organizations allowed develop products based on CAN (Controller Area Network) or other type of in-vehicle communication networks. Originally CAN is a vehicle bus standard that is designed to allow microcontrollers and devices communicate directly without a host computer.

The car manufacturers try to keep their in-Vehicle communication protocol hidden from customers. It is done in order to preserve secrets of their products. This kind of strategy kills the concept of connected car, by preventing interaction with outside world. Nowadays society requires connected car, which is synchronized with their daily life devices and their social networks. Restricting third party vendor to communicate with


J.-H. Lee and S. Pack (Eds.): QShine 2016, LNICST 199, pp. 12–21, 2017.
DOI: 10.1007/978-3-319-60717-7_2

the car by hiding and developing proprietary protocols is not the proper solution of the problem. Open source and open standards more preferred than proprietary solutions. Besides, proprietary protocols get hacked more often than open community proven secure protocols.

Dozens of research works have been done in the domain of in-Vehicle networking. In this paper we demonstrate the cryptographic protocol analysis of the recently introduced in-Vehicle secure protocol. The protocol was designed to be extremely efficient with less computational overhead. Introduced protocol has a serious flaw that puts the protocol under huge threat. There is a demonstration of the protocol analysis by applying CPN tool [6] is given. The weakness of the protocol was pointed out and demonstrated with active attacker.

Our paper is structured in a following way: in Sect. 2 some related works regarding to our work were listed out. Sect. 3 introduces CPN modelling tool and its advantages in analyzing security protocols. Sect. 4 introduces secure in-Vehicle communication protocol and its CPN model illustration. Sect. 5, demonstrate variation of replay attacks on given protocol.

## 2 Related Works

The Colored Petri Nets are recognized as a powerful tool to prove, disprove or analyze correctness of the systems, protocols and algorithms. The core of the formal analysis performed, by listing out specification of the system and creating its model. Created model can be verified using a model checking approach that consists of exploration all model states and transitions. During a process of model creation, execution and simulation the system designer can detect flaws and errors in their system design. The cause of the discovered errors and flaws can be easily seen and traced using CPN tools [6]. It gives a good prospect to find a way subsequently improve their design.

New methods to analyze cryptographic protocol using colored petri nets were introduced in [7]. They have demonstrated two new methods related to matrix description of colored petri nets to find breakable state of the net. The first one is the Acceptance Check Step (ACS) and the second one is the Matrix Analysis Step (MAS). For use case demonstration they have identified ambiguity in the wireless protocol proposed by Aziz and Diffie.

Another research work [8] demonstrated that analysis of Micali's ECS1 fair contract signing protocol. Two new attacks on ECS1 protocol have been discovered. The first attack happens due to Micali's incomplete definition on Bob's (responder's) commitment. This way intruder may claim that Bob had made commitment which he had never actually proposed. Second attack makes available to swap the initiator and responder roles in the protocol. The swapping initiator and responder's role can cause serious consequences in real life scenario.

Analysis of two OSAP and SKAP authorization protocols has been performed in [9]. The vulnerability in those protocols already been analyzed and demonstrated by [10] using ProVerif tools. The purpose of the [9] was to examine the usefulness of Colored Petri Nets and CPN Tools for security analysis. They have constructed intruder using Dolev-Yao [11] based model and generated same result as it was done in [10].

## 3 CPN Introduction

The Petri Nets are popular and well known formalism for modeling concurrency systems. Petri Net is the collection of basic elements such as places, transitions, arcs and tokens. Tokens occupy places and moves to another place through arcs when corresponding transitions enabled.

Colored Petri Nets (CPN) is an extended from original Petri Net and represents a well-known formalism for modelling concurrent protocols. CPN is applied in many areas where the concurrent and complex processes must be analyzed from architecture checking and behavior perspectives.

There are many usages of Colored Petri Nets in various domains. Kurt Jensen has written theoretical aspects of CP-Nets in [1, 2]:

- CP-Nets have a graphical representation
- CP-Nets have a well-defined semantics, which unambiguously defines the behavior of each CP-Nets
- CP-Nets are very general and can be used to describe a large variety of different systems
- CP-Nets have very few, but powerful, primitives
- CP-Nets have an explicit description of both states and actions
- CP-Nets have a semantics that builds upon true concurrency, instead of interleaving
- CP-Nets offer hierarchical descriptions
- CP-Nets integrate the description of control and synchronization with the description of data manipulation
- CP-Nets can be extended with a time concept
- CP-Nets are stable towards minor changes of the modelled system
- CP-Nets offer interactive simulations where the results are presented directly on the CPN diagram
- CP-Nets have a large number of formal analysis methods by which properties of CP-Nets can be proved
- CP-Nets have computer tools supporting their drawing, simulation and formal analysis.

### 3.1 CPN in Cryptographic Protocol Analysis

CPN is particularly good to apply for analysis of Cryptographic protocols. It can verify protocol correctness by building state space maps and by analyzing incidence matrix. In some works CPN used to verify whether any security threats exist when many instances of the protocol are executed concurrently [3].

The group of cryptographers at Queen's University and Computer Laboratory at University of Cambridge added significant research contributions to verify cryptographic and security protocols, even compute their weaknesses using CP-Nets.

There are two courses of using CP-Nets: forwards and backward analysis. Ayda and Moon stated in [4, 5], the backward state analysis has tree steps:

(1) First generate CP-Net specification for protocol
(2) Identify insecure states that may or may not occur
(3) Perform backward state analysis to test if each insecure state is reachable or not.

**State Space**
State space is one of the important features the CPN has. CPN Tools become able to inspect terminal states and identify possible deadlocks, as well as bounds on communication channels. State space is a graph that contains nodes and directed edges. Each node represent one snapshot of the CPN state, means markings positions and their values. For example first node contains initial markings positions and value information. The number of outgoing edges from that node equal to active transitions number from CPN at that concrete step. That means we can travel to each of those edges considering as proper transition got triggered. The node where the edge comes in represents new state of markings after the transition triggered. Recursively performing this operation will build up state space graph. Depend on the CPN the state space graph might get quite complex and big.

## 4 Secure In-Vehicle Communication Protocol

In this section we will introduce secure in-Vehicle communication protocol suggested by some organization (for the privacy issues we would like to classify the name of the company). The proposed protocol was registered in a patent organization for future usage, in order to provide secure in-Vehicle communication.

Protocol uses symmetric encryption, random number generator and hashing functions. Each of those used algorithms cryptographically strong and secure. It is not our goal to analyze algorithms in details or in a convergence. We assume cutting edge cryptographic algorithms are used for symmetric encryption and hashing function. We also assume random number generator has truly uniform distribution and cannot be predicted by an attacker. It is assumed that shared symmetric key to be pre-distributed between communication participants. Pre-distribution happens long before the exchange takes place and it is not considered in this protocol.

The Table 1, contains notations that are used in a formal description of secure in-Vehicle algorithm. Protocol steps are enumerated, and physical location of the operations separated with colon sign. “Serv - > ECU” refers to the transmission operation where the Server sends data to ECU (Data follows right after colon sign).

Initialization

(1) Serv: $RN_0 = G()$
(2) Serv - > ECU: $E_{Sk}(RN_0) = C$
(3) ECU: $RN_0 = D_{Sk}(C)$
(4) Serv: $K_0 = H(RN_0)$
ECU: $K_0 = H(RN_0)$

It can be easily noticed that initialization process is quite primitive and relies on pre-shared key safety and random number generator.

**Table 1.** Notations used in description of protocol.

| Messages | Notation |
|---|---|
| Sk | Secretly shared master key |
| M | Message |
| C | Cipher text (Encrypted message) |
| $RN_i$ | Random generated number on i-th step |
| $K_i$ | Session key on i-th step |
| $E_K(M)$ | Symmetric encryption of message M using key K |
| $D_K(C)$ | Symmetric decryption of message C using key K |
| H(M) | Un-keyed cryptographic hash of the message M |
| M \| C | Concatenation of messages M and C with separator |
| G() | Generate random number |
| RST | Session key resetting command |
| NULL | Refers to improper decrypted data. Means the session keys in desynchronized state |

Communication (ECU may behave as an initiator too)

(1) Serv: $RN_i = G()$
(2) Serv - > ECU: $E_{Ki-1}(M \mid RN_i) = C$
(3) ECU: $D_{Ki-1}(C) = M \mid RN_i$
(4) Serv: $K_i = H(RN_i)$
(5) ECU: $K_i = H(RN_i)$

Communication step relies on key secrecy generated at previous step. New random seed number generated and sent through the secret channel on each communication step. New random seed number us to generate session key for the next step. On the other side receiver decrypts the data with current session key. From the received data random seed number extracted and next session key derived. At the end of each step the new session keys are synchronized on both sides.

Reset (we assume desynchronization of keys happened)

(1) Serv: $RN_i = G()$
(2) Serv - > ECU: $E_{Ki-1}(M \mid RN_i) = C$
(3) ECU: $D_{Ki-1}(C) == NULL$
(4) ECU- > Serv: $E_{K0}(RST) = C$
ECU: $K_i <= K_0$
(5) Serv: $D_{Ki-1}(C) == NULL$
(6) Serv: $D_{K0}(C) == RST$
Serv: $K_i <= K_0$

Reset transaction step shows generalized version of resetting function. In a real implementation it may suggest to recover message with previous key. Previous keys are saved in key stack which has limited size (considering the ECU capabilities). Previously saved keys are extracted from the stack and tried to decrypt the received

message. When the all previously saved keys are not matched (or the stack search exhausted) then the setup returned to the initial key $K_0$. Regarding to security requirements the master key is used only at the initialization step. The key $K_0$ considered as a fixed baseline, in order to keep a master secret key away from statistical analysis.

The modelling is the fastest way to check the algorithm and all logics encompassed in it. We have followed the protocol description in details and designed CPN model of In-Vehicle Secure Communication Protocol. First we have drawn simple sketch model with just simple UNIT markings. In our opinion it is good style of drawing CPN model, because we can see overall view of the protocol and its simplified behavior. It helps approximately calculate the average number of places and transitions. After that we can start adding colored markings (new types specifically for our working domain) such as plain messages, encryption messages and session encryption messages.

The authentication process is quite straight forward where simply master key, message and random number must be involved. It was implemented by using few transitions and places (as illustrated in Fig. 1). The main body part encryption is represented as a recursive function, which is why it is packed into reusable sub-module called BodyEncryption. Controlling and monitoring parameters such as: RandomNumber, Data_Send and Data_Receive designed to be accessible and controllable from outside of the sub-module.

Detail CPN model of reusable BodyEncryption is illustrated in Fig. 2. The BodyEncryption sub-module is the core processing logic and it's reused for ECU and for HUD-Server. Both of those participants have an identical processing logic in their cores.

This protocol was designed to be strong against statistical analysis by updating session key in every exchange step. Since CAN network is quite error-prone the keychain can get broken up easily. When single exchange message is skipped by one of the participants the key mismatching state occurs. Recovery mechanism gets activated in order to solve the key mismatch issue. Recovery mechanism starts scanning keys in a backward order, by checking previous n-1 key. Key searching loop continues until proper decryption key has found. When the whole key stack is checked and key is not found, then initial key $K_0$ is used as a new start key.

The CPN model of this secure protocol revealed that design is quite error-prone to various cases. The system doesn't get locked, even if error happens with session keys, because of its automatic key recovery logic.

## 5 Multiple Replay Attacks

This protocol has multiple flaws in its design and some of them made by designer while chasing efficient secure protocol. First flaw can be detected easily by simply analyzing overall design. It applies one way authentication for initialization step. At the initialization step the recipient doesn't reply any message about success or failure of the initialization process. Not confirming the initialization step is the half of the problem, another bigger and more serious issue lies on key reset procedure.

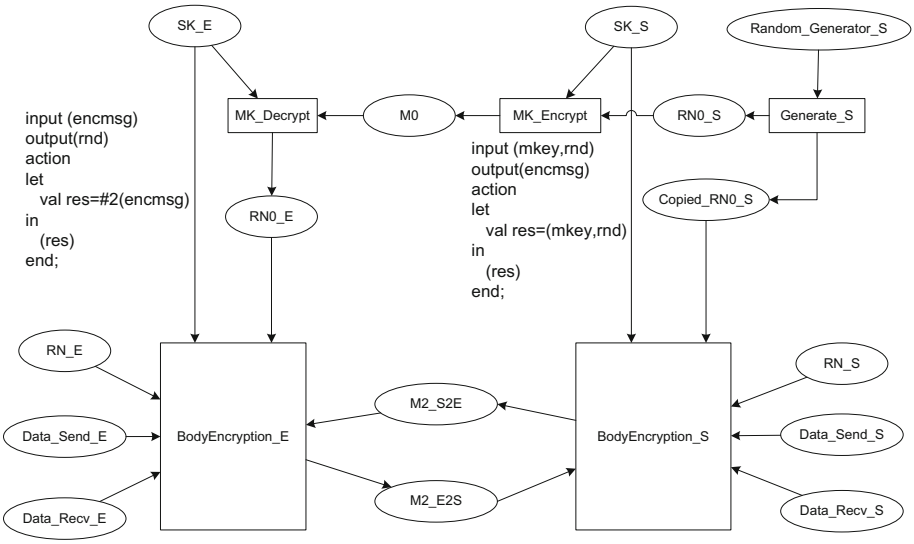


**Fig. 1.** Hierarchical model of in-vehicle secure communication protocol.

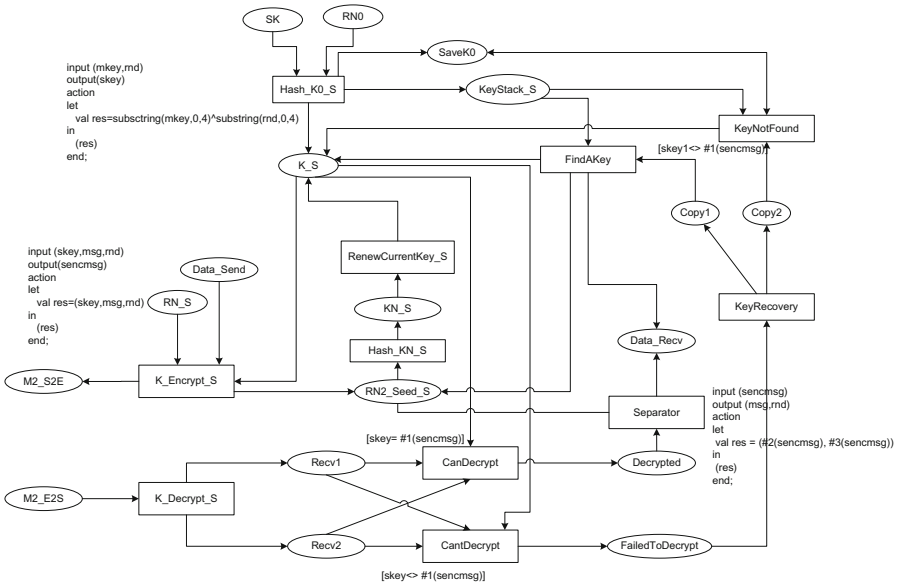


**Fig. 2.** CPN model of main encryption and error handling module (in the hierarchical model it is named as BodyEncryption submodule).

We describe two defects of the protocol by dividing this section into two parts. In first example we demonstrate how the message can be replied to make ECU perform an action. As a second example we assume there is confirmation message generated for each new message. Adding confirmation message did not exterminate the security flaw in the protocol. Main reason is that replay attack was not considered at design time.

Intruder model is used from Dolev-Yao [11], where intruder is equipped with highest imaginable strength so that all possible attacks on the protocol can be identified. We adopt the Dolev-Yao model to our domain, with CAN bus network in mind. According to the Dolev-Yao model intruder can carry out the following actions:

(1) Tapping and storage of all messages exchanged through shared bus
(2) Forwarding and blocking messages
(3) Generation of forged messages using tapped, randomly generated, hashed and encrypted messages
(4) Decryption of encrypted messages if the intruder has a matching key
(5) Intruder has the ability of normal principal, that he can take a part in the protocol and masquerade.

### 5.1 Reply Attack on Protocol Without Confirmation Message

In our example for reply attack we don't use all techniques that are available by Dolev-Yao model. Only few of them were enough to reach the goal. First of all we define our scenario where an attacker wants to perform some action on car. For example we use door lock system. In modern cars the door lock system is centralized and the center of the door lock system located at driver's door. Driver's door equipped with ECU which is connected with HUD-Server through CAN Bus.

Without diving deeply we can see that intruder can easily capture all commands initiated from HUD-Server, and later reply them back to ECU. Intruder doesn't have to exactly know the shared master key to replay the same command. The replay attack is possible because generated messages do not contain any time stamp, or ECU has too small memory to memorize all previously generated keys.

### 5.2 Reply Attack on Protocol with Confirmation Message

Even though this algorithm originally does not consider confirmation or any other measurements to prevent replay attack we will assume this functionality included (or at least we can assume this functionality added at upper layers). In this case attacker's task gets little bit complicated but still it stays in a trivial attacks class.

Replay attack on In-Vehicle Secure protocol with confirmation message illustrated in Fig. 3. Attacker just captures first initialization message and then waits for "Open Door" command. When "Open Door" confirmation command generated by ECU, attacker has to jam the network. By following key recovery protocol Server will generate "Open Door" command with basic key $K_0$. That command should be captured, so later intruder starts new session where he should just open the connection and send "Open Door" command encoded with basic key $K_0$. ECU would have no choice then just perform that command and door will open.

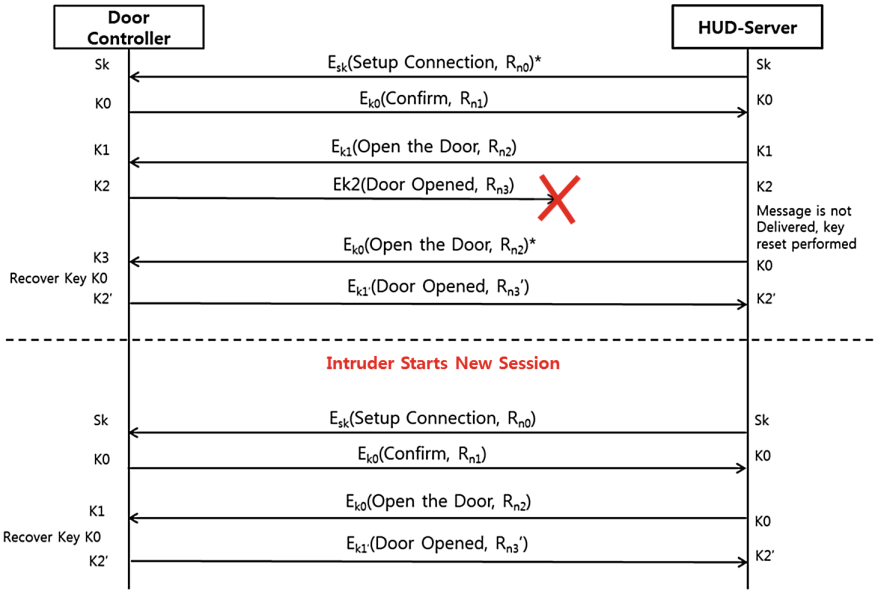


**Fig. 3.** Demonstration of replay attack when the confirmation message is enabled.

Replay attacks can be prevented by performing three handshake exchanges. To put it in a simple way, the confirmation message generated by ECU device should be re-confirmed by HUD-Server. In this case the last decision for session key would be on ECU side, which prevents replay attacker from masquerading HUD-Server. In order to send re-confirmation message an attacker should know the $K_0$. Only knowledge of $K_0$ can decrypt $Rn_1$ random seed generated by ECU. $Rn_1$ will change every time when new interaction started. The key $K_1$ should be as a baseline to recover the communication if the error happens.

## 6 Conclusion

In this paper we have analyzed In-Vehicle Secure Communication protocol. CPN Tools were used for analyzing and proving the weakness of the protocol. The protocol has been broken by applying replay attacker model. Man-In-The-Middle type of intruder model has been designed and demonstrated.

Original Protocol does not contain a confirmation message routine, that's why we have added additional confirmation message reply from the responder side. Even adding confirmation reply message has not saved the car from being attacked. Using reply attacker model we have designed real scenario with door unlocking. This flaw can be solved by generating confirmation message at initialization step. Confirmation message should contain random seed number for initialization key $K_0$. As a solution of the problem we have added re-confirmation message at the initialization step. It can prevent any kind of replay attack.

**Acknowledgments.** This work was supported by the IT R&D program of MOTIE/KEIT. [Project No.: 10060105, Development vehicle Authentication, Security, Device monitoring, Predictive Maintenance platform using OTP-based HSM].

## References

1. Jensen, K.: A brief introduction to colored petri nets. In: Workshop on the Applicability of Formal Models, Aarhus, Denmark, pp. 55–58, 2 June 1998
2. Jensen, K.: An introduction to the theoretical aspects of colored petri nets. In: Workshop on the Applicability of Formal Models, Aarhus, Denmark (1998)
3. Long, S.: Analysis of concurrent security protocols using colored petri nets. In: 2009 International Conference on Networking and Digital Society (2009)
4. Basyouni, A.M.: Analysis of wireless cryptographic protocols. Master's thesis, Queen's University Kingston, Ontario, Canada (1997)
5. Moon, H.: A study on formal specification and analysis of cryptographic protocols using colored petri nets. Master's thesis, Kwangju Institute of Science and Technology, Korea (1998)
6. www.cpntools.org – CPN Tools Homepage
7. Basyouni, A.M., Tavares, S.E.: New approach to cryptographic protocol analysis using coloured petri nets. Queen's University Kingston, Ontario (1997)
8. Sornkhom, P., Permpoontanalarp, Y.: Security analysis of Micali's fair contract signing protocol by using coloured petri nets. In: Proceedings of 9th ACIS International Conference Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD (2008)
9. Seifi, Y., Suriadi, S., Foo, E., Boyd, C.: Analysis of two authorization protocols using colored petri nets. Int. J. Inf. Secur. (2014)
10. Chen, L., Ryan, M.: Attack, solution and verification for shared authorisation data in TCG TPM. In: Degano, P., Guttman, J.D. (eds.) FAST 2009. LNCS, vol. 5983, pp. 201–216. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12459-4_15
11. Dolev, D., Yao, A.: On the security of public key protocols. IEEE Trans. Inf. Theory **29**(2), 198–208 (1983)