Physical Layer Key Generation: Securing Wireless Communication in Automotive Cyber-Physical Systems

JIANG WAN, University of California - Irvine
ANTHONY LOPEZ, University of California - Irvine
MOHAMMAD ABDULLAH AL FARUQUE, University of California - Irvine

Modern automotive Cyber-Physical Systems (CPS) are increasingly adopting wireless communications for Intra-Vehicular, Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) protocols as a promising solution for challenges such as the wire harnessing problem, collision detection, and collision avoidance, traffic control, and environmental hazards. Regrettably, this new trend results in new security challenges that can put the safety and privacy of the automotive CPS and passengers at great risk. In addition, automotive wireless communication security is constrained by strict energy and performance limitations of electronic controller units and sensors. As a result, the key generation and management for secure automotive CPS wireless communication is an open research challenge. This paper aims to help solve these security challenges by presenting a practical key generation technique based on the reciprocity and high spatial and temporal variation properties of the automotive wireless communication channel. Accompanying this technique is also a key length optimization algorithm to improve performance (in terms of time and energy) for safety-related applications constrained by small communication windows. To validate the practicality and effectiveness of our approach, we have conducted simulations alongside real-world experiments with vehicles and RC cars. Lastly, we demonstrate through simulations that we can generate keys with high security strength (keys with 67% min-entropy) with 20X reduction in code size overhead in comparison to the state-of-the-art security techniques.

CCS Concepts:
- Security and privacy → Key management; Distributed systems security; Mobile and wireless security; Symmetric cryptography and hash functions; Computer systems organization → Embedded and cyber-physical systems;

General Terms: Design, Security, Algorithms,Performance, Optimization

Additional Key Words and Phrases: Automotive Cyber-Physical Systems, Wireless Communication, Key Generation, Key Exchange, Symmetric Cryptographic Algorithm

ACM Reference Format:
DOI: 0000001.0000001

1. INTRODUCTION

Wireless technologies are widely implemented in automotive Cyber-Physical Systems (CPS) for navigation schemes (e.g., GPS) and infotainment applications such as hands-free calling, and satellite radio [Dar et al. 2010b]. As a light-weight solution to the wireless harnessing problem [Lin et al. 2014] and for its aforementioned applications, wireless technologies applied on many microcontrollers all throughout a vehicle can enable a powerful improvement in safety and comfort for people and functionality and
efficiency for automotive CPS [Dar et al. 2010a; ElBatt et al. 2006]. A notable example is the Tire Pressure Monitoring System (TPMS), which is implemented in many modern vehicles and utilizes several controllers and tire sensors to measure and display tire temperatures and pressures to passengers. Through the TPMS, passengers can understand from warning signals when to re-inflate or replace their tires, leading to evasion of unnecessary dangers.

An astounding 80% of all vehicular collisions is caused by drivers but it is clear that wireless technologies can greatly reduce the risk of driver-caused collisions and improve traffic efficiency [Weiß 2011]. In order to realize these objectives, federal agencies (e.g., United States Department of Transportation) and research organizations (e.g., Google) are developing general wireless vehicular communication protocols (V2X), which we define as either Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) [Harding et al. 2014]. With V2X communication, vehicles can perform collision detection and prevention, path control, traffic management, environmental hazard avoidance, and new forms of entertainment through Internet connection. Below, Figure 1 provides an illustration of a setting with V2X communication.

![Fig. 1. Examples of V2V and V2I Applications](image)

1.1. Motivation

With a new model of interconnectivity, both old and new V2X applications will connect traditionally isolated vehicles to each other, infrastructure, satellites and other entities through insecure wireless channels. Already, security concerns have arisen over leakage of critical information about the vehicle or the passengers and over the possibility of indirect control of the vehicle’s mechanisms [Work et al. 2008; Lawson 2008; Checkoway et al. 2011; Miller and Valasek 2014]. In fact, these security concerns date back to the mid-1990s, a time where many vehicles used Remote Keyless Entry (RKE). Researchers eventually found that RKE was vulnerable to relay attacks, where relaying a signal through simple technology could unlock and start a vehicle when its owner is away [Bono et al. 2005]. In 2005, a Texas Instrument RFID transponder implemented in millions of vehicles was found to be hackable, thus portraying another security threat [Bono et al. 2005]. Then in 2010, researchers developed an attack that captured and read TPMS communication packets from a vehicle up to 40 meters (.02 miles) away. Furthermore, they demonstrated the possibility of injecting packets into the TPMS network to trigger a fake warning signal [Ishtiaq Roufa et al. 2010a; ElBatt et al. 2006].

---

1The contents of our proposed techniques can apply to all types of automotive cyber-physical systems, however we use the average vehicle (e.g., car, truck, or any other thing used to transport people or goods) as a motivating example.
In the recent years of 2014 and 2015, researchers with support from the Defense Advanced Research Projects Agency (DARPA), have developed and demonstrated exploitable hacks regarding vehicular infotainment applications and systems like UConnect [Miller and Valasek 2014]. Their hacking demonstrations ended in the recall of many vulnerable vehicles, such as Chrysler [Miller and Valasek 2015]. An introduction of V2X communication will eventually cause similar and new security concerns. As a result, researchers such as those from the European Telecommunications Standards Institute (ETSI) are proposing the following security objectives for V2X communication: confidentiality, integrity, availability, accountability and authenticity (for more details, please see the technical reports [ETSI 2010]).

1.2. Security Challenges for V2X Communication

For the purposes of our paper, we focus on the security requirement of confidentiality. We summarize that for wireless communication in automotive CPS, messages will need to be encrypted depending on the confidentiality requirements of applications [Schütze 2011]. As a simple example, account information will need to be encrypted in financial applications like Electronic Toll Collection (ETC) [Qian and Moayeri 2008; Lawson 2008] and for cooperative pre-crash sensing [Consortium 2004]. Another major challenge for V2X communication is authentication. Since users are exposed to many dangers due to the wireless communication, it is necessary for a receiver to verify that a transmitted message was generated by a legitimate user. Recently, researchers have proposed to solve the authentication problem in automotive communication in an Ad-Hoc manner [Li et al. 2015; Chuang and Lee 2014]. However, this type of scheme requires an established secure channel for exchanging authentication information such as secret keys and identifications before communication. It is important to note that these and other security objectives apply to resource-limited (in terms of computational power, energy consumption and memory size) time-critical embedded devices (e.g., sensors, V2X) and resource-limited non-time-critical devices (e.g., infotainment) within the vehicles. Because of their important role in keeping passengers and drivers safe, we center our attention on the resource-limited and time-critical devices.

A typical automotive design needs to provide security for about 20 years or more [Schütze 2011; Wan et al. 2014], implying the necessity of a reliable and efficient cryptographic scheme to achieve some of the aforementioned security objectives. Cryptographic algorithms fall under two categories: 1) Symmetric and 2) Asymmetric. As seen in Table I, symmetric algorithms, such as the Advanced Encryption Standard (AES), have very high performance and lower energy overhead [Potlapally et al. 2006] in comparison to asymmetric algorithms, such as RSA and Elliptic Curve Cryptography (ECC). However, both of these schemes are challenging to implement on resource-limited and time-critical devices.

The major problem of using a symmetric encryption algorithm is that both parties must share a secret key to establish a secure communication. On the other hand, although asymmetric algorithms do not require a shared secret key for secure communication, they are too slow for the majority of time-critical applications and too resource-intensive in terms of computational power and memory usage [Schütze 2011; Potlapally et al. 2006; Mukherjee et al. 2010]. As a result, higher performance processors have been used to address these issues. However, using such processors (e.g., Qualcomm Snapdragon 602A for V2X) comes with a non-negligible cost (for example, the Qualcomm Snapdragon 602A may involve around $1000 or more in extra cost). Moreover, there are up to 100 Electronic Control Units (ECUs) in a modern car, and many of these ECUs are low cost micro-processors. An alternative security method would be necessary to enable V2X applications on these processors. In some of the state-of-
the-art approaches, research groups and government organizations have proposed the use of hybrid solutions to reduce overhead from the asymmetric algorithms [Schütze 2011; Scheppe et al. 2011]. In a hybrid solution, a symmetric key is generated from a Pseudo Random Number Generator (PRNG) or a Key Encapsulation Mechanism (KEM) [Hofheinz and Kiltz 2007] and exchanged through an asymmetric algorithm. Afterward, higher performance can be achieved through symmetric encryption of data.

<table>
<thead>
<tr>
<th></th>
<th>Symmetric</th>
<th>Asymmetric</th>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication</td>
<td>Message Authentication Code (MAC)</td>
<td>Digital signature</td>
<td>Digital signature on keys MAC on data</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>Encryption of data</td>
<td>Encryption of small data</td>
<td>Encrypt keys with Asym. Encrypt of data with Sym.</td>
</tr>
<tr>
<td>Performance</td>
<td>Very fast</td>
<td>Slow</td>
<td>Medium</td>
</tr>
<tr>
<td>Code size</td>
<td>Thousands of bytes</td>
<td>Thousands of bytes</td>
<td>Thousands of bytes</td>
</tr>
<tr>
<td>Key size</td>
<td>32-256 bits</td>
<td>ECC: 256-384 bits RSA: 1024-3072 bits</td>
<td>512-3072 bits for Asym. 32-256 bits for Sym.</td>
</tr>
<tr>
<td>Key management</td>
<td>Random key generation Pre-shared secret key</td>
<td>None</td>
<td>Random key generation</td>
</tr>
</tbody>
</table>

However, there are still three major limitations to the current hybrid approach: 1) It requires a key exchange session which uses an asymmetric algorithm whose lengthy computation time is generally not acceptable for safety-related applications which require a reaction time of 50 to 200 milliseconds [Schütze 2011]. 2) The hybrid solution requires an implementation of the asymmetric algorithm in the embedded devices, thus causing non-negligible memory space overhead. 3) Similar to symmetric algorithms, the hybrid solution generally relies on a Pseudo Random Number Generator (PRNG) or user-given inputs to help produce a symmetric key with high entropy. These approaches, however, cannot provide enough entropy due to high levels of predictability of the seed or user-given inputs and deterministic nature of the key generation algorithm [O’donnell et al. 2004]. For the aforementioned reasons, secret key generation and exchange are considered challenging problems for automotive wireless applications.

1.3. Related Works
To help solve this problem of developing a reliable yet efficient and fast encryption mechanism, researchers have been looking toward physical randomness as a high entropy source. As an example, researchers proposed the use of physical randomness in circuit characteristics to generate secret keys [Rostami et al. 2014; Suh and Devadas 2007]. Similarly, it is possible to exploit the physical randomness from wireless communication channel characteristics, such as the multipath-induced fading and shadow fading to generate strong secret keys. Most of the state-of-the-art theories and practical methods for generating secret keys using physical characteristics of the wireless channel (or the physical layer) have been proposed within just the last decade but have not been applied to the automotive CPS environment [Bloch et al. 2008; Mathur et al. 2008; Ye et al. 2010; Zeng et al. 2010; Wang et al. 2011; Ren et al. 2011; Jana et al. 2009; Patwari et al. 2010; Zan et al. 2013].

2Entropy (more specifically Shannon Entropy) can be used as a quantified value of randomness for a set of bits.
The success of generating secret keys based on the wireless channel’s physical properties depends on three properties: 1) Reciprocity of the radio wave propagation, 2) Temporal variation, and 3) Spatial variation in the wireless environment (see details in Section 2). Besides most of the theoretical works [Bloch et al. 2008; Mathur et al. 2008], some practical implementations for sensor network applications [Ren et al. 2011; Jana et al. 2009; Azimi-Sadjadi et al. 2007] have been performed and rely on the Multiple-Input and Multiple-Output (MIMO) approach or collaborations among multiple wireless nodes to create secret keys with high entropy. Work in [Zan et al. 2013] has also provided an implementation for V2X applications. However, it mainly focuses on comparing their algorithm with other key generation algorithms and modeling the spatial and temporal variations of the automotive wireless channel. Moreover, the authors do not consider practical challenges such as abiding by real-time requirements for safety-critical V2V applications and optimizing their algorithm to its highest potential in terms of resources such as time, energy, and memory. Lastly, other types of physical layer security have been discussed in [Mukherjee et al. 2010] but have not yet been applied to the automotive CPS domain to enable efficient and reliable security protocols as we have done.

In summary, solving the limitations of the above-mentioned state-of-the-art approaches to secure wireless communication in automotive CPS poses the following challenges:

1. Finding a reliable high entropy source to generate secret keys for symmetric cryptographic algorithms.
2. Designing a reliable solution for the management of symmetric secret keys.
3. Optimization of the solution and key size in terms of performance.

1.4. Overview of Contributions

To address the challenges, we propose a novel secured communication scheme for automotive wireless communication. The foundation of the proposed scheme is the technique from our published paper in the International Conference on Cyber-Physical Systems [Wan et al. 2016], which generates symmetric cryptographic keys based on a pre-shared key (PSK) from the physical randomness of the automotive wireless channel under tight memory and performance budgets. The biggest advantage of this technique is that it solves the key generation and exchange problem at the same time, which means it can generate the PSKs with high entropy while eliminating the costly requirements of the asymmetric algorithms for the key exchange process. Moreover, it can also replace the asymmetric algorithm for exchanging secret keys and identification for authentication purposes. In comparison to our previous paper, the novel contributions of the proposed scheme include: 1) Protocol algorithms to support the proposed PSK generation technique: including Scenario Mapping, PSK Optimization, and Cryptographic Key Derivation; 2) New simulation results corresponding to these algorithms; and 3) New experimental results, including more detailed comparisons between our technique and RSA and ECC. To the best of our knowledge we are the first to demonstrate, through realistic automotive modeling, simulation and experiments, that higher levels of entropy and performance may be obtained from the moving and changing environment to practically generate symmetric secret keys for automotive CPS wireless communication. The contributions of this paper are as follows:

1. Automotive wireless communication system models (Section 2) including the channel and attack models from a security perspective.

A PSK is also known as a shared secret (a random bit string known only to a pair of communicating parties).
A secured communication scheme for automotive wireless communication which includes:
(a) A wireless channel-based PSK generation technique (Section 3) that incorporates the results from optimization to generate a PSK for two communicating parties in automotive communication.
(b) A PSK length optimization technique (Section 4) under timing constraints based on the scenario of vehicular communication.
(c) Cryptographic key derivation methods (Section 5) to convert the PSK into a suitable key for encryption purposes.

Simulations and real world experiments (Sections 6 and 7) to demonstrate the practicality of our proposed PSK optimization and generation techniques.

For a full visual description of our secured automotive wireless communication scheme, please see Figure 2.
time, we can assume that the estimated channel gain is the same for both A and B: \( H'_{A \rightarrow B}(t) \approx H'_{B \rightarrow A}(t) \). However, from the eavesdropper’s side, the estimated channel gains \( H'_{A \rightarrow E}(t), H'_{E \rightarrow A}(t), H'_{B \rightarrow E}(t) \) and \( H'_{E \rightarrow B}(t) \) will be independent of \( H'_{A \rightarrow B}(t) \) and \( H'_{B \rightarrow A}(t) \), if the eavesdropper is a few wavelengths [Ye et al. 2010] away from the legitimate wireless channel. Utilizing this concept, the channel gain \( (H'_{A \rightarrow B}(t) \) and \( H'_{B \rightarrow A}(t)) \) may be used to extract PSK bits (see our technique in Section 3) for security purposes.

The wireless communication channel gain varies over time due to temporal or spatial variations in the environment. Typically, the channel may be modeled with a fast fading model or a slow fading model depending on the changing speed of the environment [Simon and Alouini 2005]. For automotive CPS, if there exists a velocity difference between two communicating automotive wireless nodes, we use a fast fading model (temporal variation), otherwise, we use a slow fading model (spatial variation).

2.1.1. Fast Fading. For the fast fading model, we assume a Rayleigh fading channel [Simon and Alouini 2005] which is suitable for modeling vehicular wireless communication [Simon and Alouini 2005] in an urban driving profile. The Rayleigh fading channel models the Doppler shift effect [Simon and Alouini 2005] due to the different speeds between two communicating wireless nodes. In this model, the channel gain \( H \) should abide by the following Probability Distribution Function (PDF):

\[
PDF_H(H, \sigma) = \frac{H}{\sigma^2} e^{-H^2/(2\sigma^2)}
\]

where \( \sigma \) is an environment-related parameter. Due to the Doppler shift effect, \( H \) only remains constant within the coherence time [Simon and Alouini 2005] \( T_c \) (see the following Equation).

\[
T_c \approx \frac{0.423}{f_d}
\]

here, \( f_d \) is the maximum Doppler frequency. During automotive wireless communication between A and B, \( f_d \) may be decided by the speed difference of the two communicating vehicles \( \Delta V \) as shown below:

\[
f_d = \frac{\Delta V}{c} f_0 \\
\Delta V = |V_A - V_B|
\]

where \( c \) is the speed of light and \( f_0 \) is the carrier frequency.

This model reflects that the channel changes roughly every time interval of \( T_c \). In other words, the higher the \( \Delta V \) is, the more frequent the channel changes and the quicker a channel-based PSK may be generated. Extracting information from the channel gain \( H \) to generate a secret bit must be done within a given time period, \( T_c \). Otherwise, the changes in the channel after \( T_c \) may cause mismatches between the generated PSKs of the communicating automotive wireless nodes.

2.1.2. Slow Fading. When the relative speed between the communicating automotive wireless nodes is low, \( \Delta V \approx 0 \), the fast fading model will not work. Therefore, we use a general slow fading model for the wireless communication. In a slow fading channel, the gain remains correlated in time if the channel does not move over a certain distance. This distance is defined as the coherence length \( L_{coher} \). On the other hand, the

\[^4\text{In a wireless communication system, the coherence time is the time duration over which the channel impulse response is considered to be invariant.}\]
model assumes that if the channel moves further than $L_{\text{coher}}$, the channel gain will become independent of the previous channel gain. Therefore, considering the velocity of the vehicle $V$, we may calculate the coherence time for a slow fading channel as follows:

$$T_c \approx \frac{L_{\text{coher}}}{V}$$  \hspace{1cm} (6)

Similar to the fast fading channel model, the slow fading channel also changes roughly every time interval, $T_c$. As demonstrated with Equation 6, $L_{\text{coher}}$ is decided by the environment. In other words, the higher the $V$ is, the more frequently the channel physically changes. The time varying channel gain for a slow fading model follows the log-normal distribution as shown below:

$$PDF_H(H, \sigma) = \frac{1}{H\sigma\sqrt{2\pi}} e^{-\frac{\ln(H)^2}{2\sigma^2}}$$  \hspace{1cm} (7)

2.2. Attack Model

In this paper, we consider a classic non-intrusive wireless attack model where the attacker tries to decipher the message by eavesdropping on packets from the legitimate wireless channel through a separate wireless channel. We assume that the attacker can capture all the wireless packets sent through the legitimate wireless channel and the attacker knows all the information about the communication system including modulation/coding techniques and cryptographic algorithms. Therefore, in such a scenario, if the attacker can get the related pre-shared or cryptographic key, the system security requirements will be broken. As a result, we define attack strength $\text{AttackStr}$ as a rate at which the attacker can employ a given amount of computing hardware resources to evaluate a number of keys within a period of time. We note that intrusive attacks are not considered in this paper since they typically require the use of highly expensive and impractical devices and are challenging to implement on specific vehicles in real-time scenarios. Further, we note that no knowledge about the attacker is necessary (such as channel state information [Mukherjee et al. 2010]) for our algorithm as the attacker will generally be farther than a wavelength (approximately 5 centimeters for the 802.11p automotive communication protocol [Stibor et al. 2007]) away from the legitimate wireless channel. Notice that, in the case of V2I communication, we also assume that the infrastructure is typically physically protected so that the attackers may not eavesdrop the messages within a few wavelengths from the infrastructure.

2.3. Security Strength Evaluation

Security strength indicates the amount of work (number of operations) an attacker would need to do in order to break a cryptographic algorithm or system. According to the National Institute of Standards and Technology (NIST) standard, an algorithm or system is defined to have “N-bits security strength” when it requires an attacker to perform around $2^N$ operations to break the algorithm or system [Barker et al. 2006]. In our case, the security strength is equivalent to the number of bits in the pre-shared key, which is the basis for the cryptographic key.

Nonetheless, in order to directly measure the randomness and security strength of the PSK, we choose to use the concept of min-entropy [Holcomb et al. 2009]. As a worst case estimation, min-entropy provides the lower bound of randomness. Let $K$ be the set of all possible PSKs randomly generated, the min-entropy is defined as follows:

$$H_{\infty} = H_{\min} = -\log(\max_{k\in K} Pr[K = k])$$  \hspace{1cm} (8)

where, $Pr[K = k]$ is the probability of generating PSK $k \in K$. 

ACM Transactions on Cyber-Physical Systems, Vol. 1, No. 1, Article 1, Publication date: January 2017.
Thus, we model the security strength, $\text{Security}_{str}$, of a cryptographic algorithm or system using the average min-entropy on each bit of the key as follows:

$$\text{Security}_{str} = H_{\text{min}} / \text{Key size}$$

where, $\text{Key size}$ is the size of the key and $\text{Security}_{str}$ is a value ranged from 0 to 1 in the unit of bits. For example, a 128-bit key with $\text{Security}_{str} = 0.5$ bit will have 64 bits of min-entropy.

3. PRE-SHARED KEY GENERATION

We present our PSK generation algorithm with a V2V wireless communication example shown in Figure 3. In this example, both Alice and Bob are driving, where Alice’s vehicle ($A$) is communicating with Bob’s vehicle ($B$). Assume the driving velocities for $A$ and $B$ is $V_A$ and $V_B$, respectively and the velocity difference between these two moving vehicles is $\Delta V$. The coherence time $T_c$ of the communication channel between $A$ and $B$ may be estimated using Equation 4 and Equation 5. Now, if $A$ and $B$ want to generate a PSK with size of $PSK_{size}$, they need to exchange a set of pre-defined probe signals (can be any kind) to evaluate the randomness of the channel gain $H$ using Equation 3. In order to have a low mismatch rate, they must exchange each probe signal within the Coherence Time ($T_c$) interval. Meanwhile, in order to keep bits of the generated PSK uncorrelated to each other, the time interval defined as $\tau_{step}$, between exchanging each probe signal should be no less than $T_c$. Notice that, as long as the sender $A$ and receiver $B$ share the same $\tau_{step}$, the process of exchanging pre-defined signals is naturally synchronized. In this paper, we assume there exists a pre-defined $\tau_{step}$ for both $A$ and $B$. Otherwise, since knowing $\tau_{step}$ will not help the attackers to get the generated PSK, we suggest that one solution would be that the sender $A$ and receiver $B$ may make an agreement on $\tau_{step}$ through public communication before the PSK generation process.

![Fig. 3. Physical Layer PSK Generation for a V2V Scenario](image-url)
3.1. Algorithm Pseudocode and Description

A predefined group of probe signals with a group size $G_{size}$ is sent for evaluating the channel randomness. After the probe signals are exchanged, a set of measured Received Signal Strength (RSS) values is used to generate secret key bits on each side. We then implement a mismatch checking step to remove mismatching bits. During this step, the sender and receiver will publicly exchange the indexes of the probe signals which are used for generating secret bits, in $PSK_{idx}$ and remove the mismatched indexes. Notice that the exchange is public and the attacker may easily get $PSK_{idx}$. However, the attacker will not be able to figure out the generated bits because only the sender and receiver share the RSSI values of the probe signals. Once a set of matching bits is generated, the set’s size must be greater than or equal to the required PSK length, $L_{PSK}$. If the set of bits is not long enough, the whole process reiterates until it is. The pseudocode of the wireless channel-based PSK generation algorithm is presented in Algorithm 1.

**ALGORITHM 1**: A Wireless Channel-Based PSK Generation Algorithm for Automotive CPS

- **Input**: Measured Signal Strength: $RSS$
- **Input**: Sample Time Step: $\tau_{step}$
- **Input**: Group Size: $L_G$
- **Input**: Threshold Parameter: $\alpha$
- **Input**: Required Pre-Shared Key Length: $L_{PSK}$
- **Output**: Generated Pre-Shared Key: $PSK$

L = 0; $PSK = 0$; $RSS_{set} = \emptyset$; $RSS_{filtered} = \emptyset$; $PSK_{idx} = \emptyset$

while $L < L_{PSK}$ do

for $i = 1$ to $L_G$ do

$RSS_{set} = RSS_{set} \cup RSS_i$

Wait($\tau_{step}$);

$RSS_{filtered} = RSS_{set} * H_{highpass}(t)$;

$MeanValue = \text{Average of } RSS_{filtered}$;

$Var = \text{Variance of } RSS_{filtered}$;

$Th_{up} = MeanValue + \alpha * Var$;

$Th_{lo} = MeanValue - \alpha * Var$;

foreach $RSS_j \in RSS_{filtered}$ do

if $RSS_j > Th_{up}$ then

$PSK = (PSK << 1) + 0$;

$L = L + 1$;

Record $j$ in $PSK_{idx}$;

else if $RSS_j < Th_{lo}$ then

$PSK = (PSK << 1) + 1$;

$L = L + 1$;

Record $j$ in $PSK_{idx}$;

Exchange $PSK_{idx}$;

Remove mismatch bits from $PSK$;

return $PSK$;

Lines 3-5 take $(L_G \times \tau_{step})$ time to collect all Received Signal Strength (RSS) values from the wireless channel. Line 6 filters the low frequency parts of the collected RSS values with a high pass filter defined by its impulse frequency response $H_{highpass(t)}$. 

ACM Transactions on Cyber-Physical Systems, Vol. 1, No. 1, Article 1, Publication date: January 2017.
The filtered signal values $RSS_{filtered}$ contains all the information that we need to extract the secret bits. Lines 7-10 calculate the thresholds used for generating bits from the received RSS values. As proposed by [Premnath et al. 2014], we use two thresholds. Every RSS value greater than the upper threshold $Th_{up}$ is considered as 1 and every RSS value less than the lower threshold $Th_{lo}$ is considered as 0. Any value in between $Th_{up}$ and $Th_{lo}$ is discarded. The thresholds $Th_{up}$ and $Th_{lo}$ are calculated by the equations in Line 9 and Line 10, respectively, based on the mean and variance values of the collected RSSs. Additionally, we use $\alpha$, which is a configurable parameter for reducing the bit mismatches due to the existence of noise. If the signal-to-noise ratio of the channel and the transmitters is low, $\alpha$ may need to be set as a higher value to reduce the number of mismatches; otherwise, $\alpha$ may be chosen to be a lower value to improve the performance of the algorithm. Lines 11-19 check all the collected RSSs and generate a PSK, $PSK$, with length, $L$. Notice that, Line 15 and Line 19 also record the index of all suitable RSSs for generating secret bits. The indexes in $PSK_{idx}$ from the two communicating automotive wireless nodes are exchanged in Line 20. Then, in Line 21, $PSK_{idx}$ is used to remove all mismatching bits. Finally, if $L >= L_{PSK}$, a PSK is generated among both communicating parties; otherwise, the algorithm will reiterate.

4. PRE-SHARED KEY LENGTH OPTIMIZATION

Our objectives in creating a scheme to secure wireless communication for the automotive domain includes determining an optimal PSK length according to a scenario. We define a scenario as an instance of communication between an automotive cyber-physical system and another entity (can either be an automotive or non-automotive entity) within a specific setting such as the streets or highway. For example, a scenario can be a vehicle communicating with a tolling device in a highway. In this scenario, the vehicle and tolling device will need to generate a PSK within a small amount of time. [Schütze 2011; Dar et al. 2010b]. For these reasons, it is necessary to generate an optimal PSK length for our technique to finish under timing constraints with as much security strength as possible. More specifically, the following characteristics determine a scenario: 1) Types of communicating parties, 2) Location, 3) Fading model (slow or fast) and 4) Coherence time/length. From these details, we can determine the scenario and key lifetime to generate an optimal PSK length for our proposed physical key generation technique. For the following sections, we provide the scenarios we defined and used as motivating examples for our simulations. These can be altered and extended according to designers.

4.1. Scenario Mapping

4.1.1. V2V Scenarios. In V2V, if a scenario is an emergency, a communication session would last less than a second, but if it is not, a session could last to many minutes for traffic efficiency purposes. Therefore, a V2V scenario is based on the severity of the situation as summarized by the following: 1) Emergency Avoidance (milliseconds), 2) Emergency Detection (seconds) and 3) Traffic Efficiency (minutes) [Schütze 2011]. It can be possible to detect the severity of a V2V scenario by the amount of time that has passed since the start of the communication session. To do this, we assume that all initiated V2V communication sessions are first categorized as emergency avoidance to prevent unexpected collisions. After many key refreshes, we can relax the scenario time constraints to generate a longer PSK for higher security strength and longer lifetime.

4.1.2. V2I Scenarios. V2I communication scenarios include communication with roadside units, tolling stations, and traffic lights among others. Infrastructure is generally
motionless, implying that to compute the maximum time of communication, or total latency, we need to take into consideration the overall distance which the vehicle will need to cover before it goes out of the infrastructure’s communication range. In general, the highest ideal range for realistic communication is approximately .5 miles (750 meters) to .6 miles (1000 meters). [Stibor et al. 2007; Belanovic et al. 2010; Dar et al. 2010b]

Computing the optimal PSK length for a scenario therefore requires us to calculate the total latency (see next subsection) and the coherence times corresponding to all possible relative velocities. Although we assume general relative velocities such as approximately 0-45 Miles per Hour (mph) for a street scenario and 0-75 mph for a highway scenario [Stibor et al. 2007], it is important to note that the actual relative velocity is computed before PSK length optimization by the vehicles’ embedded devices.

4.1.3. Total Latency. Total latency is the assumed limit to how long two entities in vehicular communication will communicate. For V2V in general, we set the total latency of communication between two vehicles to be 200 milliseconds [Schütze 2011], since we always first assume a collision detection and avoidance scenario. On the other hand, we can subdivide the scenarios of V2I communication based on the setting (streets or highway), where the velocity differences in V2I highway scenarios are vaster than those of V2I street communication. By considering the maximum velocities of each type of scenario, we choose highway and street communication total latencies to be approximately 10 seconds and 40 seconds, respectively.

4.1.4. Fading Models. The decision of the fading model is important in the proposed scheme. We make a reasonable assumption that a modern automotive system includes various sensors, such as speed radar, to provide an estimation of the relative speed between other vehicles. As a result, we propose a possible solution for selecting the fading model as follows: For velocity differences greater than 5 mph, we use the fast fading

```
ALGORITHM 2: Scenario Mapping

Input: Scenario: Scen
Input: Fading Model: Model
Output: Coherence Times: CoherTimes
Output: Total Latency: TotalLat
1 CoherTimes = ∅
2 TotalLat = getTotalLatency(Scen)
3 if Model == FastFadingModel then
4     VelocityDiffs = getVelocityDifferences(Scen)
5     foreach VelocityDiff_i ∈ VelocityDiffs do
6         Compute coherence time, CoherTime_i, using VelocityDiff_i
7         CoherTimes = CoherTimes ∪ CoherTime_i
8 if Model == SlowFadingModel then
9     Velocities = getVelocities(Scen)
10    foreach Velocities_i ∈ Velocities do
11       Compute coherence length, CoherLength_i, using AsphIndex, SpeedOfLight, and Bandwidth
12       Compute coherence time, CoherTime_i, using Velocity_i and CoherLength_i
13       CoherTimes = CoherTimes ∪ CoherTime_i
14 return CoherTimes, TotalLat
```

ACM Transactions on Cyber-Physical Systems, Vol. 1, No. 1, Article 1, Publication date: January 2017.
model for our scenario mapping technique. From the range of velocity differences, we compute corresponding coherence times using equations 4 and 5 in Section 2.1.1.

For low velocity differences such as 5 mph or under, we require the slow fading model to compute the coherence times and corresponding optimal PSK lengths. We suggest that for two unmoving communicating parties or an emergency avoidance scenario where the parties have low relative velocity (e.g., approximately 0-5 mph), it would be better to use a stored pre-distributed key to implement the symmetric encryption algorithm instead.

When our algorithm does use the slow fading model, we use equation 6 provided in Section 2.1.2 to calculate the coherence times from the coherence lengths and scenario mapping values. To compute the coherence lengths, we use the index of refraction of asphalt (1.635), speed of light, and bandwidth of the channel (5.9 Ghz) [Stibor et al. 2007]. The pseudocode of our scenario mapping algorithm is given in Algorithm 2.

Moreover, we want to emphasize that the fading model itself is not the contribution of this paper. They can be dynamically updated using more detailed models to improve the key generation performance. And the decision of the fading model can also be agreed between two communication nodes through public communication before the key generation process.

4.2. Algorithm Pseudocode and Description

We can now optimize the PSK length using the scenario mapping function and attack model. For safety-critical and resource-limited devices, we optimize using the timing constraint: LifeTime (the total scenario-based latency or the time until a PSK needs to be regenerated to prevent an attacker from computing it). Additionally, we use a parameter called Fract, a dimensionless input which provides the designer an option to adjust the timing constraints based on their unique requirements, such as preventing key generation from taking away valuable communication time (since key generation requires packet exchanges). For the purpose of evaluation, we set it to be .2 but increase it to .4 for 5-10 mph and .7 for 0-5 mph. As an example, with Fract = .2, the algorithm will compute an optimal length such that the estimated key generation time will be within 1/5th of the lifetime. Consequently, our algorithm will ascertain out of a set of PSK lengths, the most viable optimal length that satisfies this time constraint (LifeTime * Fract). In this case, since a chosen PSK length may not meet the length requirement of the encryption method, we will extend the PSK into valid cryptographic keys through methods discussed in Section 5. Our PSK length optimization algorithm is detailed in Algorithm 3 and simulation results for our method are provided in Section 6.

In the algorithm, we perform Binary Search to discover the optimal PSK length in terms of generation time under lifetime constraints. We thus require minimum and maximum lengths in terms of bits (which we specify as 1 and 128) to set the range of our search space. As the velocity difference (or velocity for slow fading model) increases, we expect that within the time interval of LifeTime * Fract, the optimal length also increases. This is because a higher velocity difference (velocity) directly enables a higher PSK bit generation rate. In addition to determining the optimal PSK length, the algorithm also produces the effective and generation times and energy overhead values. Notably, this algorithm can also be customized to compute the optimal PSK length under energy constraints (which can be provided as an extra constraint in Line 16). In general, the energy constraints can be determined based on the power consumption of the vehicle’s embedded device when computing a single PSK bit. The optimal PSK length will serve as input for the physical layer PSK generation technique specified in the following section.
ALGORITHM 3: Algorithm for PSK Length Optimization

Input: Minimum Length: $PSKLenMin$
Input: Maximum Length: $PSKLenMax$
Input: Power to Generate a PSK Bit: $Power$
Input: Attack Strength (128-Bit Keys per Second): $AttackStr$
Input: Scenario: $Scen$
Input: Fraction of the Lifetime: $Fract$
Input: Fading Model: $Model$
Output: PSK Generation Energy Values: $EnergyVals$
Output: Optimal PSK Lengths: $OptPSKLens$
Output: Optimal Lifetimes: $OptLifeTimes$
Output: Optimal PSK Generation Times: $OptPSKGenTimes$

Current PSK Length: $CurrPSKLen = 0$
{
CoherTimes, $TotalLat$} = ScenarioMapping($Scen$, $Model$

foreach $CoherTime_i \in \text{CoherTimes}$ do

Perform Binary Search to find optimal PSK length

while $(PSKLenMax >= PSKLenMin)$ do

$CurrPSKLen = \text{Mid}(PSKLenMin, PSKLenMax)$
$PSKGenTime = CoherTime_i * CurrPSKLen$
$PSKGenEnergy = Power * PSKGenTime$
$LifeTime = \min(2^{\frac{128-CurrPSKLen}{AttackStr, TotalLat}})$
if KeyGenTime <= (LifeTime $\times$ Fract) then

$OptPSKLen = CurrPSKLen$
$OptLifeTime = LifeTime$
$OptPSKGenTime = PSKGenTime$
$Energy = PSKGenEnergy$
$PSKLenMin = CurrPSKLen$

else

$PSKLenMax = CurrPSKLen$

$OptPSKLens = OptPSKLens \cup CurrPSKLen$
$OptLifeTimes = OptLifeTimes \cup LifeTime$
$OptPSKGenTimes = OptPSKGenTimes \cup PSKGenTime$
$EnergyVals = EnergyVals \cup Energy$

return $OptPSKLens, OptLifeTimes, OptKeyGenTimes, EnergyVals$

5. CRYPTOGRAPHIC KEY DERIVATION

For cryptographic purposes, we need to generate a cryptographic key with length in accordance to a valid existing encryption scheme. For V2V and V2I communication, we will assume that the encryption scheme is AES-128, as it is a fast and efficient standard symmetric encryption scheme (on the order of 9 microseconds to encrypt and decrypt 60 bytes). [Schütze 2011] For AES-128, the key length must be 128 bits; however, since we are concerned with safety applications in V2X communication, we want to provide our algorithm with an optimized PSK length to minimize overhead. This means that the length of the PSK may be less than 128 bits and therefore requires an additional key derivation step.

To convert the PSK into a longer, cryptographically secure key, we propose the use of either the HMAC Key Derivation Function (HKDF) [Krawczyk 2010] or the Merkle-Damgard bit padding algorithm [Goldwasser and Bellare 2001]. Given the key length and the total latency, one solution to choose between the key derivation functions is
provided in Algorithm 4. The HKDF takes in the desired key length, KeyLen, the pre-shared key, PSK, and some mutual data as the salt, Salt, such as past traffic information (e.g., environmental, lane changes, speed changes, etc.), as input, and it outputs a cryptographic key of KeyLen bits. If there is no prior data exchanged between each other, the two parties can use null as the Salt. The resulting key is created by the HKDF concatenating partial results from a one-way hash function, such as the HMAC-SHA256, on both of the inputs. Putting the inputs into the function together, we can run CryptoKey = HKDF(PSK, Data, KeyLen), where CryptoKey is the resulting 128-bit cryptographic key. On the other hand, it is possible to implement the Merkle-Damgard bit padding algorithm on a PSK with insufficient length to convert it into a 128-bit key. The Merkle-Damgard algorithm pads a 1, then successive 0s, and finally the length of the original PSK to the end such that the length is equal to a desired length. This method is simpler and faster than the HKDF, although it does not necessarily produce strong cryptographic keys.

We want to note that the short key is only used for a short session where 1) the expiration times of a message are small, 2) the key generation algorithms are typically not suitable, and 3) confidentiality is not required but integrity is. Despite the short seed length, the key refresh rate derived from the key optimization algorithm (see Algorithm 3) will help prevent attackers from easily computing the key while it is being used. By the time the attacker figures out the key and decrypts the message after it has been sent, the communication scenario may have changed considerably (for example, broadcasting of real-time traffic information, and V2V communication for emergency purposes) and the decrypted messages are no longer valid afterwards (since they typically have expiration times associated with them). Further, it is important to note that for scenarios where the timing requirements are not strict or the communication session is relatively long, we can generate keys with longer lengths within acceptable time. When there is a scenario where the timing requirement is strict and confidentiality is required, we can generate a key with suitable length using another key generation technique [Barker et al. 2006]. Our approach and traditional approaches can coexist with one another on the automotive system. In cases where other approaches lack in performance, our approach can be used, and vice-versa. Thus, by introducing our approach, we believe that automotive designers can design a new type of automotive system such that security is optimized in terms of both performance and security strength.

**Algorithm 4: Algorithm for Cryptographic Key Derivation**

<table>
<thead>
<tr>
<th>Input: Scenario: Scen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: PSK: PSK</td>
</tr>
<tr>
<td>Input: Salt: Salt</td>
</tr>
<tr>
<td>Input: Desired Key Length: KeyLen</td>
</tr>
<tr>
<td>Output: Cryptographic Key: CryptoKey</td>
</tr>
</tbody>
</table>

1. if \( \text{SizeOf}(PSK) < 128 \) and \( \text{getTotalLatency}(Scen) == 2 \) then
2. \[ \text{CryptoKey} = \text{Merkle-Damgard}(PSK, KeyLen) \]
3. else if \( \text{SizeOf}(PSK) < 128 \) \&\& \( \text{getTotalLatency}(Scen) > 2 \) then
4. \[ \text{CryptoKey} = \text{HKDF}(PSK, Salt, KeyLen) \]
5. else
6. \[ \text{CryptoKey} = \text{Truncate}(128, PSK) \]
7. return CryptoKey
6. SIMULATION RESULTS

6.1. Pre-Shared Key Generation

In this section we first evaluate our wireless channel-based PSK generation technique without optimization, implying that generating a PSK is equivalent to generating the cryptographic key. We used MATLAB [MathWorks 2014] to simulate the automotive wireless channel with the following parameters. The average driving speed is set to 37 mph and the coherence length for slow fading is set to 20 meters, or about .01 miles, for an urban environment [Weiß 2011]. The simulation evaluates the key generation time with respect to the relative speed between two communicating nodes (0 to 75 mph in our setup). Moreover, the simulation is conducted with respect to 6 different key sizes (56, 112, 128, 168, 192, 256 bits) proposed by the security standards from NIST [Barker and Roginsky 2011]. The summarized simulation setup is presented in Table II.

<table>
<thead>
<tr>
<th>Tested Key Length (bits)</th>
<th>56, 112, 128, 168, 192, 256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relative Velocity Range (mph)</td>
<td>0 to 75</td>
</tr>
<tr>
<td>Average Velocity (mph)</td>
<td>37</td>
</tr>
<tr>
<td>Signal to Noise Ratio (dB)</td>
<td>80</td>
</tr>
<tr>
<td>Coherence Length (mi)</td>
<td>.01</td>
</tr>
<tr>
<td>Group Size (bits)</td>
<td>10</td>
</tr>
</tbody>
</table>

As presented in Figure 4, our key generation algorithm has negligible performance (10 to 100 milliseconds) overhead when the relative speed is high due to the fast fading of the wireless channel. This implies that our algorithm can be suitable for various V2X applications and scenarios. On the other hand, for the scenario where the relative speed between two nodes is around zero such as intra vehicle communication, our simulation results show a longer generation time (around 1 to 2 minutes). However, compared to the lifetime of the key, which is typically several hours to even months in these scenarios, several minutes can also be considered as negligible. Although in some cases, several seconds of overhead for generation is not acceptable (e.g., safety related applications), our wireless channel-based key generation algorithm can be optimized using the schemes in Section 4.

We additionally conducted simulations to confirm the independence of two generated keys from two different automotive wireless communication channels to demonstrate that the attacker cannot easily retrieve the key by eavesdropping. The simulation setup is presented in Figure 5. Three vehicles (with driving profiles) are modeled and connected using the developed wireless channel models. Two wireless channel models are instantiated in the simulation, where one connects the vehicle models with Drive Profile 1 and Drive Profile 0 to each other, and the other connects the vehicle models with Drive Profile 1 and Drive Profile 2 with each other.

For each relative speed in mph and key size, as specified in Table III, we run the simulation 100 times to generate two vectors of keys from two wireless channels at the same time. Then, we calculate the Pearson’s correlation coefficient [Lee Rodgers and Nicewander 1988] between these two vectors. The calculated correlation results are presented in Table III. From the simulation results, we can observe that all the correlation results are close to zero (the highest correlation value is just 0.0392). These results demonstrate the low correlation of keys generated from the channels of two
vehicles connected to the same target through wireless communication, thus implying that the attacker cannot retrieve the key generated from the legitimate wireless channel by this method.

Table III. Correlations of the Generated Keys

<table>
<thead>
<tr>
<th>Key Size</th>
<th>56 bits</th>
<th>112 bits</th>
<th>128 bits</th>
<th>168 bits</th>
<th>192 bits</th>
<th>256 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 mph</td>
<td>0.0102</td>
<td>0.0121</td>
<td>0.0132</td>
<td>0.0207</td>
<td>0.0305</td>
<td>0.0233</td>
</tr>
<tr>
<td>12 mph</td>
<td>0.0271</td>
<td>0.0053</td>
<td>0.0361</td>
<td>0.0221</td>
<td>0.0337</td>
<td>0.0125</td>
</tr>
<tr>
<td>25 mph</td>
<td>0.0264</td>
<td>0.0132</td>
<td>0.0026</td>
<td>0.0125</td>
<td>0.0177</td>
<td>0.0283</td>
</tr>
<tr>
<td>37 mph</td>
<td>0.0176</td>
<td>0.0177</td>
<td>0.0056</td>
<td>0.0293</td>
<td>0.0334</td>
<td>0.0268</td>
</tr>
<tr>
<td>50 mph</td>
<td>0.0039</td>
<td>0.0236</td>
<td>0.0167</td>
<td>0.0392</td>
<td>0.0147</td>
<td>0.0244</td>
</tr>
</tbody>
</table>

Fig. 4. Simulation Results of Our Key Generation Overhead

Fig. 5. Simulation of Generating Two Secret Keys at the Same Time
6.2. Pre-Shared Key Length Optimization

In this section, we evaluated our PSK length optimization using four example scenarios: V2V/V2I Streets and Highway. From the scenario mapping, we generated the appropriate total latencies, velocity differences or velocities by using the suitable fading models, maximum velocities and incremental values. Furthermore, from these data, we computed the coherence and lifetimes assuming an attacker with low budget and corresponding strength of $2.3 \times 10^7$ (128-bit keys per second) [Ecrypt ii]. Finally, we determined the optimal PSK lengths for each scenario to allow PSK generation to occur within a small fraction of the lifetime. Our results are shown in Figures 6 and 7 and optimization parameters are provided in Table IV.

Additionally, as shown in Figures 8 and 9 we provide the PSK generation time in accordance to optimized PSK lengths. From the figures, it is quite apparent that higher velocity difference implies a lower key generation time. For V2I scenarios, the optimal PSK length of 128 bits can be generated within milliseconds to seconds. As for V2V scenarios, where time is of the essence to detect and prevent collisions, it is apparent that the key generation times must be lower in comparison (up to 100X smaller) to those of the V2I scenarios. Nonetheless, it is possible to create a PSK with 5-50 bits and convert it using our cryptographic key derivation techniques specified in Section 5.

Table IV. Scenario Mapping Data

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Total Latency (s)</th>
<th>Velocity Range (mph)</th>
<th>Increment (mph)</th>
</tr>
</thead>
<tbody>
<tr>
<td>V2V Streets</td>
<td>0.2</td>
<td>0-45</td>
<td>3</td>
</tr>
<tr>
<td>V2V Highway</td>
<td>0.2</td>
<td>0-75</td>
<td>5</td>
</tr>
<tr>
<td>V2I Streets</td>
<td>40</td>
<td>0-45</td>
<td>3</td>
</tr>
<tr>
<td>V2V Highway</td>
<td>10</td>
<td>0-75</td>
<td>5</td>
</tr>
</tbody>
</table>

Fig. 6. Optimized Key Length for Fast Fading Model
Fig. 7. Optimized Key Length for Slow Fading Model

Fig. 8. Optimized Key Generation Time for Fast Fading Model

Fig. 9. Optimized Key Generation Time for Slow Fading Model
7. EXPERIMENTAL RESULTS
Going further than simulation, we conducted real world experiments to validate the proposed physical layer key generation technique and treat the PSK as the cryptographic key to evaluate and demonstrate the practicality of our solution using Algorithm 1.

7.1. Remotely Controlled Car Environment
In our first experiment, we used a system made up of three Remotely-Controlled (RC) cars and Raspberry Pis connected via Bluetooth. As presented in Figure 10, we mounted the Raspberry Pi systems on top of the RC cars. On each Raspberry Pi board, we attached Bluetooth dongles (via USB) to establish the wireless communication. One of our objectives was to confirm nearly zero correlation between generated keys from different channels within a short distance, but longer than a few wavelengths (for Bluetooth, the wavelength is approx .125 meters). Therefore, we mounted two Bluetooth dongles on Car 1 (as shown in Figure 10) to establish two wireless communication channels between Car 1 and Car 0, and Car 1 and Car 2. During runtime, all the Received Signal Strength (RSS) values from each Bluetooth dongle were collected by a computer through separate WiFi channels (as shown in Figure 10). For each Bluetooth communication channel, we collected RSS values from the communicating nodes. Thus, in total there were four sets of RSS values collected from all Bluetooth dongles. Although for this experiment we used a computer to execute the key generation algorithm and analyze its results, we can also easily implement the same key generation algorithm in the Raspberry Pis.

![Fig. 10. RC Car Experiments Setup](image)

We consider the experimental environment with RC cars as a slow fading one because the cars move at low speeds (less than 5 mph) and within a distance of 10 meters (about .006 miles) from each other in open areas with few moving objects around them. 200 samples of the collected RSS values are presented in Figure 11. From the
results, we can easily observe that the RSS values collected at Car 1 and Car 0 for the wireless communication between Car 1 and Car 0 are highly correlated with each other (shown in red lines). The same results are also found for the wireless communication between Car 1 and Car 2 (shown in blue lines). These results clearly show the reciprocity characteristic of the wireless communication channel. Moreover, we have found that even with very short distances, the generated RSS values from two different wireless communication channels have nearly zero correlation, thus supporting the assumption that “an attacker that is several wavelengths away will experience different wireless channel characteristics, and therefore cannot obtain or predict the keys.” Table V shows the generated 64 bits of keys based on the collected 200 samples of data. Notice that, we use 50 as the probe signal group size for the key generation algorithm in this experiment.

Table V. Generated 64-bit Keys from the RSS Values

<table>
<thead>
<tr>
<th>Generated 64-Bit Keys</th>
</tr>
</thead>
</table>
| Car 1 from Car 0 1100000110000000_0000000100000110_
|                                        |
| Car 0 from Car 1 1100000110000000_0000000100000110_
|                                        |
| Car 1 from Car 2 0000001111111111_1111000000000011_
|                                        |
| Car 2 from Car 1 0000001111111111_1111000000000011_
|                                        |
7.2. Automobile Environment

In order to further validate the practicality of our algorithm, we also performed experiments in real driving scenarios. To do so, we implemented the Bluetooth protocol in our customized applications to acquire RSS values of the wireless channel in real time.

As presented in Figure 12, we placed the mobile devices in two vehicles and recorded the RSS values from both vehicles while driving to generate keys. We demonstrate the RSS values received from both sides of the mobile devices during a period in Figure 13. We can observe that there exists several mismatched signals in Figure 13, this is primarily because Bluetooth communication is not stable between the two fast vehicles resulting in some loss of RSS data. However, our Algorithm 1 already considers these mismatches and handles them well. In this experiment, we would like to note that the RSS value sampling period is 10 milliseconds due to the limitations of the Bluetooth devices (mobile phone and laptop in this experiment). Due to the low sampling rate, each RSS sample will always be obtained after each coherence time, thus prohibiting our algorithm from reaching its full speed.
The experiments are conducted based on three relative speeds of 20, 10, and 2 mph with both vehicles moving in the same direction. Throughout the experiment, we collected the RSS values and generated six keys with unique lengths (see Figure 14). We want to note that sampling takes the majority of time and our key generation algorithm’s execution time is negligible (constant).

![Fig. 14. Experimental Key Generation Time at Different Speeds](image)

### 8. EVALUATION

In this section, we compare our works with the state-of-the-art hybrid cryptographic algorithms [Moharrum and Al-Daraiseh 2012; Schütze 2011] to evaluate the security strength, performance and code size overhead for automotive wireless communications.

#### 8.1. Security Comparison

We compare the security strength of our algorithm’s generated keys to those produced by other techniques. We evaluate and compare the security strength using the proposed average min-entropy as the Key Performance Indicator (KPI). Traditional wireless sensor communication uses pre-distributed keys [O’donnell et al. 2004] for their practicality (a simple key management scheme) in achieving real-time communication. However, since the pre-distributed keys and associated algorithms are predictable, the pre-distributed key approaches have little to no entropy [O’donnell et al. 2004]. In comparison to the traditional approach, approaches that use PUFs⁵, such as the SRAM-PUF [Holcomb et al. 2009], can generate keys with high average min-entropy.

To estimate the average min-entropy of our key generation algorithm, we ran our simulation 12800 times to generate $100 \times 2^8 = 12800$ number of 8-bit keys. Based on the collected keys, we calculated the probability $Pr_{max}$ of the key with the highest likelihood and applied this $Pr_{max}$ to Equation 8. Figure 15 shows the resulting average min-entropy of our technique in comparison with other well-known techniques such as pre-distributed keys, Latch-PUFs, DFF-PUFs, and SRAM-PUFs. Note that our algorithm can generate keys with security strength close to that of some of the best PUF-based approaches (up to 67% average min-entropy for 8-bit keys⁶). Although

---

⁵A Physical Unclonable Function (PUF) is a function based on physical characteristics that are practically impossible to be duplicated.

⁶According to [van den Berg 2012], the average min-entropy increases with the respect to the size of the key.
some of the PUF-based approaches (e.g., SRAM-PUF) can generate keys with higher average min-entropy (since the number of 0 and 1 bits tend to be around the same), our algorithm has the advantage of generating keys by directly accessing the communication channel without needing a special physical process such as SRAM rebooting (for SRAM-PUFs). While the average min-entropy (67%) is not as high as some of the PUF-based approaches, it can be potentially increased by adding hardware or algorithm improvements.

Fig. 15. Estimated Average Min-Entropy Results Comparison

8.2. Performance Overhead Comparison

From the performance point of view, we know that the wireless channel-based key generation algorithm has the advantage of not needing the time-consuming key exchange step of asymmetric and hybrid techniques. Thus, we compare our algorithm’s generation time to the execution time of two of the most popular asymmetric cryptographic algorithms (RSA and ECC [Gura et al. 2004]) used in hybrid solutions [Schütze 2011]. The comparison is conducted given two different NIST security strength (80 and 112 bits) requirements (please refer back to Section 2.3 for more details). To evaluate the algorithm’s performance, we use the generation time for two different relative velocities (2 mph and 20 mph) collected from our experiments (presented in Section 7.2). To obtain the execution times of RSA and ECC, we have implemented ECC and RSA on the Raspberry Pi platform, which has a 1.2GHz 64-bit ARMv8 processor, for two cases: 1) there is no public-private key generation and pre-installed public keys are simply exchanged, and 2) new public-private key pairs are generated and exchanged. The first case refers to the possibility that only pre-installed public-private key pairs are used and exchanged but no key generation occurs in RSA/ECC. On the other hand, the second case refers to the possibility that the public-private key pairs are updated with a key generation step and then exchanged in RSA/ECC. According to the best of our knowledge, the key generation step in asymmetric cryptography is important [Barker et al. 2006] in order to prevent major security problems such as leakage or eventual reconstruction of the private keys. In fact, in many security protocols involving public-private key pairs for asymmetric encryption, a method for key generation is specified or required.

The performance overhead and code size overhead comparison results are provided in Table VI. The first two columns from the left under Performance Overhead correspond to the two cases where no key generation occurs but the key exchange, sign and verify steps do occur. They reveal that our algorithm is much slower than both RSA and ECC given the assumption that a public-private key pair is already established and only key exchange, signing and verifying occur. The results in the two adjacent columns to the right under Performance Overhead correspond to the case where a key
generation step occurs before key exchange, signing and verifying in RSA/ECC. These results demonstrate that our approach performs considerably closer to and better than RSA/ECC. Given different scenarios in V2X communication where the communication session may last for several seconds or minutes, our Algorithm 3 will be able to find a proper PSK length (which should be small in most scenarios) so that the security requirements will be met while the key generation time is negligible. The performance of our key generation approach for higher relative speeds mainly depends on how many PSK bits need to be generated and has a linear relationship with security strength.

In summary, although considerably slower than RSA/ECC key management where there is no key generation step, our approach replaces both the public-private key pair generation and exchange steps of asymmetric cryptographic approaches and can run faster than even RSA for certain scenarios. Specifically, in comparison to RSA/ECC, our approach has the following advantages: 1) the advantage of optimizing the key generation time based on the scenario, whereas RSA/ECC will take approximately the same amount of time for each public-private key pair generation step in any type of scenario, and 2) the advantage of a dynamic key generation technique based on physical randomness, whereas RSA/ECC may use a pre-installed and static public/private key pair that is not updated by key generation (or very infrequently). The current results demonstrate that our proposed technique can be a fair alternative method for V2X communication from a performance perspective.

### 8.3. Code Size Overhead Comparison

In order to evaluate the overhead from the memory size point of view, we also compare the code size of our algorithm to sizes of implemented RSA and ECC algorithms. For a fair comparison, we cross-compiled the code of our proposed key generation algorithm to make it suitable for the same 8-bit processor and to get a valid code size. As shown in Figure VI, our algorithm code in comparison is 10X smaller than ECC code and is 20X smaller than RSA code [Gura et al. 2004]. Additional code, including that of the key length optimization algorithm and the cryptographic key derivation methods, are also negligible in size and easily programmable onto the constrained devices.

<table>
<thead>
<tr>
<th>Security Strength</th>
<th>Performance Overhead (Seconds)</th>
<th>Code Size Overhead (Bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>80 bits</td>
<td>[Raspberry Pi] [Raspberry Pi]</td>
<td>[Raspberry Pi] [Raspberry Pi]</td>
</tr>
<tr>
<td></td>
<td>RSA (sign &amp; verify) RSA (key setup)</td>
<td>ECC (sign &amp; verify) ECC (key setup)</td>
</tr>
<tr>
<td></td>
<td>0.02 0.34 0.4 0.52 1.725 0.95</td>
<td>6292 3682 331</td>
</tr>
<tr>
<td>112 bits</td>
<td>[Raspberry Pi] [Raspberry Pi]</td>
<td>[Raspberry Pi] [Raspberry Pi]</td>
</tr>
<tr>
<td></td>
<td>RSA (sign &amp; verify) RSA (key setup)</td>
<td>ECC (sign &amp; verify) ECC (key setup)</td>
</tr>
<tr>
<td></td>
<td>0.16 13.1 0.9 1.16 2.415 1.33</td>
<td>7736 4812 331</td>
</tr>
</tbody>
</table>

### 9. CONCLUSION

We have presented a physical layer symmetric cryptographic key generation methodology that exploits the randomness of the automotive wireless communication channel. We have also provided an optimization algorithm for the technique in terms of length, time and/or energy according to V2V/V2I scenarios. Our technique is a low-cost solution, in terms of performance and code size, to help solve the challenging key exchange problem and confidentiality requirements in automotive wireless communication. As demonstrated by our results, the proposed algorithm can generate keys with up to 67% average min-entropy. In addition, our proposed technique can achieve 20X code size reduction in comparison to the state-of-the-art hybrid cryptographic algorithms.
In summary, we developed a simple yet powerful proof of concept for a practical wireless channel-based symmetric cryptographic key generation technique that can coexist with existing state-of-the-art methods to improve the security and performance of automotive CPS.

REFERENCES


Physical Layer Key Generation for Automotive Cyber-Physical Systems


Received XXX; revised XXX; accepted XXX