Short Paper: MVSec: Secure and Easy-to-Use Pairing of Mobile Devices with Vehicles

Jun Han[†‡], Yue-Hsun Lin[†], Adrian Perrig[‡], Fan Bai[§]
[†]Carnegie Mellon University [‡]ETH Zurich [§]General Motor Research
{junhan, tenma}@cmu.edu adrian.perrig@inf.ethz.ch fan.bai@gm.com

ABSTRACT

With the increasing popularity of mobile devices, drivers and passengers will naturally want to connect their devices to their cars. Malicious entities can and likely will try to attack such systems in order to compromise other vehicular components or eavesdrop on privacy-sensitive information. It is imperative, therefore, to address security concerns from the onset of these technologies. While guaranteeing secure wireless vehicle-to-mobile communication is crucial to the successful integration of mobile devices in vehicular environments, usability is of equally critical importance. With *MVSec* , we propose novel approaches to secure vehicle-to-mobile communication tailored specifically for vehicular environments. We present novel security protocols and provide complete implementation and user study results.

Categories and Subject Descriptors

C.2.0 [**General**]: Security and Protection; C.2.1 [**Network Architecture and Design**]: Wireless Communication

Keywords

Secure key agreement; smartphone security; vehicle security

1. INTRODUCTION

With the proliferation of wireless devices using Wi-Fi and Bluetooth technologies, security of their communication is a vital concern as numerous real-world attacks have been reported [14]. Insecure wireless communication may allow attackers to eavesdrop or launch Man-in-the-Middle (MitM) attacks, impersonating legitimate communicating devices.

Efforts to eradicate such attacks have inspired many research proposals as well as industrial solutions, namely to provide secure pairing between the devices by "bonding" them to establish a secure channel. However, it is still significantly difficult for human users to easily determine which devices are being paired because of the invisible nature of wireless communication. Hence, researchers propose demonstrative identification, which affirms to the human user which devices are actually communicating leveraging out-of-band channels. [3].

However, many naive solutions attempting to establish such secure pairing for any two devices introduces a tradeoff. In many cases, increasing security leads to decreased usability, which becomes a significant hindrance for wide adoption of the technology by the general public. On the other hand, decreased usability may cause a security breach in these protocols. This is exemplified by the use case scenario when a user tries to pair her phone with a friend's phone using Bluetooth. The state-of-the-art solutions require the user to either copy a passkey displayed on one device to the other, compare two passkeys displayed on both devices, or to enter a hard-to-guess passkey on both devices. However, the security of such protocols often rely on the passkey not being repeated or easy-to-guess, requiring the users to input hard-to-guess passkeys to guarantee the protocol security [10]. These designs, however, lead to multiple problems in practice. Many devices actually display a repeated and/or easy-to-guess passkeys (e.g., 000000, 123456, etc.) [17]. Also, many users tend to make fatal mistake of inputting easy-to-guess passkeys [15].

In this paper, we delve into a specific problem of vehicular environments. The proliferation of smartphones coupled with emerging smarter vehicles allows constant exchange of sensitive information over wireless communication. For example, different automotive manufacturers and smartphone companies established Car Connectivity Consortium (CCC) and have formed *Mirror Link*, a standard for integrating smartphones and the vehicles to enable access to the phones using car's control, display, and speakers [6]. In addition to pairing with personal cars, we expect more frequent pairing use cases for widely deployed rental car services – both traditional and short-term rental cars (e.g., Zipcar).

Unfortunately, coupling of smartphones and vehicles introduces a new avenue of potential attacks if the wireless channel is not secured. Although launching such attacks may not seem plausible at a first glance, they are certainly within the realm of possibility especially for high-value targets (e.g., celebrities, politicians, etc.) that provide more incentives for the attackers. Furthermore, such targets are more likely to drive luxury vehicles that embrace next-generation vehicle-to-mobile convergence systems. Are current cars effectively protected from remote attackers attempting to compromise vehicular components? Can we be convinced that the sensitive information in our vehicles is not being maliciously transmitted to attackers in other nearby cars on the road, or in parking lots connected via Bluetooth or Wi-Fi?

WiSec'14, July 23–25, 2014, Oxford, UK.
Copyright 2014 ACM 978-1-4503-2972-9/14/07 ...$15.00.
http://dx.doi.org/10.1145/2627393.2627400.

To address these problems, we present *MVSec* , the first secure key agreement scheme tailored specifically for vehicular environments, providing strong security guarantees and easy usability. MVSec leverages out-of-band channels such as sound or light as its communication medium because commodity hardware such as LED, ambient light sensor, speaker, and microphone are readily available in cars and/or smartphones. MVSec allows a user, typically the driver, to simply press a button on each device (the car and the phone) to initiate the protocols. For the protocols that leverage sound, all the user needs to do is to simply verify that both the car and the intended mobile device emit a short beep. Similarly, for protocols that leverage light emission, the user simply needs to place the mobile device in the glove compartment for a short amount of time. We present detailed explanation of more protocols and their security analysis in §4.

This paper makes the following **contributions**. We provide (1) a description of MVSec vehicle-to-mobile pairing protocols which leverages different cryptographic schemes based on various out-of-band channels readily available in commercial vehicles and mobile devices; (2) an implementation of the MVSec protocols on Android smartphones; and (3) experimental results to demonstrate the usability of MVSec, as it requires minimal user involvement.

2. PROBLEM DEFINITION

This section presents the goals we plan to achieve given the constraints, lists the assumptions we hold, and discusses the attacker model.

Goals. The main goal of MVSec is to present a complete system that provides a secure and usable communication between the car and the smartphone. If an attacker is present and launches an attack, it will be clear to the user that an error has occurred, so that the user can immediately abort the pairing process. The main properties MVSec tries to achieve are the following. MVSec achieves **secrecy** by allowing the driver's phone and the car to hide information from unintended devices. It also achieves **authenticity** and **integrity** by allowing the driver's phone and the car to validate that unaltered data arrived from the claimed sender. MVSec also achieves **demonstrative identification** by enabling the user to explicitly be aware of which devices are actually communicating via the wireless communication.

Constraints. We also categorize some of the constraints that pose challenges in achieving the aforementioned goals. The phone and the car initially do not share any prior secret, nor depend on any Trusted Third Party (TTP) for exchanging the secret. In addition, MVSec incurs minimal hardware cost by leveraging available hardware commonly installed in today's cars and smartphones to communicate via out-of-band (OOB) channels (discussed further in §4).

Assumptions. We make the following assumptions to achieve the aforementioned goals. We assume that the OOB channel *does not require user diligence*. This is a necessary assumption to ensure high usability. We also assume that there is *no malware* on the vehicle or mobile device. If there is malicious code on the mobile device, a pairing protocol will no longer securely establish a shared secret.

Attacker Model. We now present the attacker model by describing the attacker goals and capabilities.

Attacker Goals. The goals of the attacker is to break the aforementioned security properties, namely to break secrecy and authenticity of vehicle-to-mobile communication.

Attacker Capabilities. We assume that the attacker can perform both passive and active attacks. A passive attacker can perform attacks without actively participating in the protocol, such as eavesdropping. An active attacker follows a Dolev-Yao attacker model who are able to perform various types of attacks in addition to eavesdropping – data injection attacks, denial-of-service, man-in-the-middle (MitM), etc. In this paper, we concentrate on defending against the MitM attack.

3. RELATED WORK

Many researchers have investigated the problem of securely pairing two devices that do not share prior secret key. One of the main challenges in secure pairing, however, is to provide usability while guaranteeing security.

Wireless solutions such as Bluetooth or Wi-Fi have standards that attempt to provide a secure exchange of credentials (e.g., Bluetooth Secure Simple Pairing (SSP) [8] and Wi-Fi Protected Setup [1]). We illustrate as an example the details of one of the SSP protocols called *numeric comparison*. This protocol consists of two phases in performing a secure pairing. In the first phase, a pair of devices exchange public keys (e.g., Diffie Hellman). In the second phase, both devices perform verification on the received public keys by requiring the user to verify if the displayed numbers on both devices are identical. Once the user performs a successful verification, the devices then establish a secure connection. However, Kuo et al. [10] highlight that large attack surfaces for these specifications exist, and provide recommendations to improve usability. For example, the security of the *numeric comparison* method depends on the displayed number to be hard to guess and unrepeated. However, in many products, manufacturers are not careful in their implementations, and cause potential security vulnerabilities.

Different research proposals are suggested to achieve secure pairing while preserving usability. One approach is to leverage a visual channel. McCune et al. propose Seeing-is-Believing (SiB) [12], a solution that allows two smartphones to securely exchange each other's public keys using QR codes and phone cameras. SiB, however, is not well suited for a vehicular setting because it requires extra hardware such as cameras, which is not present in vehicles. SiB also requires user diligence as the users need to actively take pictures of the QR code.

4. MVSec PROTOCOLS

This section presents the overview of the MVSec protocols, discusses the OOB channel selection, and then delves into the protocol details. The main goal of MVSec is to allow a user to securely pair his/her smartphone with a vehicle such that an attacker will not successfully launch MitM attacks.

To achieve this goal, we first need to overcome the challenge of providing *demonstrative identification*, to ensure that the vehicle and the intended smartphone are in fact communicating with each other. We leverage out-of-band (OOB) communication channels as a solution. Different from the in-band channels used by the devices, e.g., Wi-Fi or Bluetooth, an OOB channel is a separate communication medium between the communicating devices (e.g, humans, light, sound, vibrations, etc.).

4.1 Out-of-band Channel Selection

MVSec leverages two types of OOB channels for the protocols described in detail in §4.2. They are categorized into *strong* and *weak* OOB channels.

Strong OOB Channel. A strong OOB channel guarantees both *secrecy* and *authenticity.* We select **light** in a vehicle's closed glove compartment as the strong OOB channel because it provides both of these security properties. We assume that the glove compartment does not leak any light signal, thus provides secrecy. This channel also provides authenticity because only the vehicle will emit light signals. This is because no other device is inside the compartment as the driver first verifies that other devices are not placed inside the compartment during protocol execution.

In addition to considering the security properties, we choose light to conform to the assumptions made in §2. The OOB channel needs to (1) be readily available in vehicles and smartphones today in order to be easily deployable, and (2) provide high usability, i.e., the OOB channel needs to have a relatively fast data rate and should not require user diligence nor annoy the users. We define relatively fast data rate to be faster than the OOB channel used as baseline case, which is manual human input (explained further in §5). This OOB channel allows such usability because the only task that the user performs is to press a button on both the vehicle and the smartphone, and place the smartphone inside the glove compartment. After waiting for a few seconds, during which the vehicle transmits signals via blinking lights to the smartphone, the pairing process successfully completes.

Weak OOB Channel. A weak OOB channel provides only *authenticity.* We select **sound** signals as the weak OOB channel. This channel provides authenticity because a user can easily identify that the sound beeps are originating only from the intended devices (e.g., vehicle and driver's smartphone). If an unintended device beeps, the user simply aborts the protocol. We assume that the beeps are sufficiently long and loud enough for the user to easily identify the origin of the beeps. We assert that this is a realistic assumption, because smartphone users generally distinguish who's phone is ringing when (s)he hears a phone ring. We also use sound signals because of the ubiquitous deployments of microphones and speakers in vehicles and smartphones.

4.2 MVSec Protocol Details

This section describes the MVSec protocols that leverage light and sound signals as strong and weak OOB channels, respectively. We present the underlying cryptographic primitives of these protocols.

MVSec-I: Protocol using EKE

```
1. $User$                               :Presses start buttons on A and B.
                                         Places B in the glove compartment.
2. $A \overset{Light}{\Longrightarrow} B$  :$K_s$ where $K_s \xleftarrow{R} \{1,0\}^\ell$
3. $A \xrightarrow{BT} B$               :$\{g^a\}_{K_s}$; B decrypts $\{g^a\}_{K_s}$ with $K_s$;
   $B$                                  :Computes shared key $K' = (g^a)^b$.
4. $B \xrightarrow{BT} A$               :$\{g^b\}_{K_s} || M_{K'}(n_A)$ where $n_A = H(\{g^a\}_{K_s})$
   $A$                                  :Decrypts $\{g^b\}_{K_s}$;Computes shared key $K = (g^b)^a$;
                                         $M_{K'}(n_A) \overset{?}{=} M_K(H(\{g^a\}_{K_s}))$;
                                         Aborts if verification failed.
5. $A \xrightarrow{BT} B$               :$M_{K'}(n_B)$ where $n_B = H(\{g^b\}_{K_s})$
   $B$                                  :$M_{K'}(n_B) \overset{?}{=} M_K(H(\{g^b\}_{K_s}))$;
                                         Aborts if verification failed.
```

Figure 1: MVSec-I using light as the strong OOB channel with $\ell = 20$.

MVSec-I: Protocol leveraging a strong OOB channel. The first key agreement protocol leverages light as a strong OOB channel. This protocol makes use of the Encrypted Key Exchange (EKE) [4] and is depicted in Figure 1. A conventional EKE scheme allows two participating entities to use a shared low-entropy password to derive a temporary shared key that can be used to authenticate the key exchange messages. We use a variant of the EKE scheme by treating a short shared secret K_s (20 bits) as a low entropy password. K_s is first transmitted via the light signal in Step 2. In Steps 3 and 4, both the vehicle and the smartphone transmit their DH public keys encrypted with K_s. Then the vehicle and the mobile device also performs key confirmation in Steps 4 and 5.

MVSec-II: Protocol using SAS with Hash

```
1.  $User$                                :  Presses start buttons on A and B. Aborts if devices
                                             other than A or B beep during execution.
2.  $A \xrightarrow{BT} B$                :  $C_A = H(g^a)$.
3.  $B \xrightarrow{BT} A$                :  $C_B = H(g^b)$.
4.  $A \xrightarrow{BT} B$                :  $g^a$
5.  $B$                                   :  $C_A \overset{?}{=} H(g^a)$ verifies $g^a$ and abort if verification fails.
                                             Computes shared key $K = (g^a)^b$.
    $B \xrightarrow{BT} A$                :  $g^b$
6.  $A$                                   :  $C_B \overset{?}{=} H(g^b)$ verifies $g^b$ and abort if verification fails.
                                             Computes shared key $K' = (g^b)^a$.
7.  $A \overset{Sound}{\Longrightarrow} B$ :  $SAS_A = [H(K')]_\ell$.
    $B$                                   :  $SAS_B = [H(K)]_\ell$;
                                             $SAS_A \overset{?}{=} SAS_B$; aborts if verification fails.
8.  $B \overset{Sound}{\Longrightarrow} A$ :  $SAS_B$.
    $A$                                   :  $SAS_B \overset{?}{=} SAS_A$; aborts if verification fails.
Key confirmation (check $K' \overset{?}{=} K$)
9.  $A$                                   :  $n'_A \xleftarrow{R} \{1,0\}^\eta$.
    $A \xrightarrow{BT} B$                :  $n'_A || M_{K'}(n'_A)$
10. $B$                                   :  $n'_B \xleftarrow{R} \{1,0\}^\eta$.
    $B \xrightarrow{BT} A$                :  $n'_B || M_K(n'_A || n'_B)$
11. $A$                                   :  $M_K(n'_A || n'_B) \overset{?}{=} M_{K'}(n'_A || n'_B)$;
                                             abort if confirmation fails.
    $A \xrightarrow{BT} B$                :  $M_{K'}(n'_B)$
12. $B$                                   :  $M_{K'}(n'_B) \overset{?}{=} M_K(n'_B)$;
                                             abort if confirmation fails.
```

Figure 2: MVSec-II using sound as the weak OOB channel with $\ell = 20$ and $\eta = 256$ (HMAC-SHA3).

MVSec-II: Protocol leveraging a weak OOB channel. MVSec-II uses sound as the weak OOB channel in Figure 2. This protocol leverages *Short Authenticated Strings (SAS)* [16, 13, 11] which uses commitment/decommitment schemes prior to transmitting the short hash comparisons for verification. This approach is preferred over a naive approach of sending short hash values over the weak OOB channel for verification. The reason is that attackers may be able to launch attacks to find hash collisions. The vehicle and the smartphone transmit their commitment messages in Steps 2 and 3. These commitments are hash of their DH public keys. They reveal the public keys to each other in Steps 4 and 5. After verifying the public keys by comparing the hashes, both parties generate negotiated DH key K and K' in Steps 5 and 6. To confirm the correctness of negotiated DH key, two parties exchange the SAS messages in Steps 7 and 8 via the weak OOB channel. The SAS messages here are truncated to only 20 bits. Then, both parties verify whether the received SAS and the transmitted SAS matches. If successful, the two parties perform key confirmation as depicted in Steps 9 to 12.

4.3 Discussion of MVSec protocols.

In MVSec-I, the vehicle and the smartphone only share a short secret key K_s, because the low data rate of the OOB channel renders transmission of a longer key (e.g., 128-bit AES key) impractical. We do not use this key directly for data encryption or authentication, but rather as a short term shared secret. Otherwise, an attacker may perform brute-force attacks to derive this key.

The light in a glove compartment is a unidirectional OOB channel. In order to perform mutual authentication, we provide secrecy in addition to authenticity, hence a strong OOB channel. However, the sound channel is bidirectional, and therefore, does not require the additional secrecy property.

We analyze the security of the proposed protocols and how they successfully defend against MitM attacks. We also verify the protocols using AVISPA [2], a state-of-the-art automated security protocol validation tool. [1]

5. IMPLEMENTATION

We demonstrate working MVSec protocols using the Android platform. We use two Motorola Droid 1 phones running Android 2.2.3 (Froyo) - one to simulate the car and the other to represent the driver's smartphone respectively.

5.1 MVSec Pairing Walk-Through

We now provide a walk-through of the MVSec pairing protocols by describing our implementation prototypes leveraging both the weak and strong OOB channels.

Weak OOB Channel. First, both devices prompt the user with instructions and a "Pair Now" button on the car. Once the user initiates the pairing process, the device simulating the vehicle (car for short) will start transmitting light pulses by varying the light intensity levels of the screen. §5.3 also provides implementation detail. The phone will capture the varying light intensity levels in this step. Then the car and the phone will exchange messages over the in-band channel (i.e., Bluetooth) to complete MVSec-I protocol.

Strong OOB Channel. Once the user presses the "Pair Now" button, the two devices will initiate pairing messages over the in-band channel. Then the two devices transmit each other's SAS messages over the OOB channel (sound). As soon as the car finishes emitting the beep, the phone starts beeping, and the car listens. In §5.2, we present a detailed description of the encoding and decoding of the sound pulses. After the SAS messages have been exchanged, the two devices complete MVSec-II protocol by exchanging the key confirmation messages over the in-band channel.

5.2 Audio Channel

MVSec leverages sound as a weak OOB channel for the following reasons. First, the SAS messages only need to be authenticated, but not require to be secret. The audio channel provides authenticity because the driver can easily determine the source device of the sound beeps inside the car - i.e., whether the beep is originating from the car speakers and his intended mobile device, as opposed to other unintended devices (e.g., passenger smartphone). Second, the necessary hardware are already available in the car and the phone, which satisfies the constraints mentioned in §2. This is because all cars and phones have speakers and microphones.

In order to transmit 20 bits of the SAS message, we first encode the data into eight different frequencies, allowing 20 bits to be encoded to 8 pulses. It takes roughly 800 ms to transmit a pulse (including the pause), so it takes roughly 5.6 seconds to transmit all 20 bits of data. For example, when the transmitter transmits 0x93759 as the SAS message, it is first encoded the message to '2233531' in base 8. On the receiver's side, we leverage Android's AudioRecord class to record the sound signal. Once the signal is recorded, we filter the signal by applying Goertzel algorithm [7, 5] for the eight target frequencies. We use eight frequencies evenly distributed from 900Hz to 1600Hz. The aggregate of the filtered frequencies represented by their magnitude squared (mag^2), is depicted in Figure 3. Each spike represents the pulse that correlates to a base 8 number. To finalize the decoding phase, we process the pulses by applying a sliding window technique to the mag^2 values. The sliding window algorithm is triggered when the mag^2 value exceeds a certain threshold, *th*. Upon triggering the sliding window algorithm, we check to see if the mag^2 value exceeds *th* within a certain window size, *wnd*. If the value exceeds *th*, we increment a counter until it exceeds the detection threshold, *dth*. We then classify this window as a legitimate sound pulse. Using empirical analysis, we set *wnd*=4000, *th* = 40, and *dth*=200. The described processing increases the detection accuracy, and reduce false positives, and successfully decodes the pulses to the correct '2233531'(0x93759).

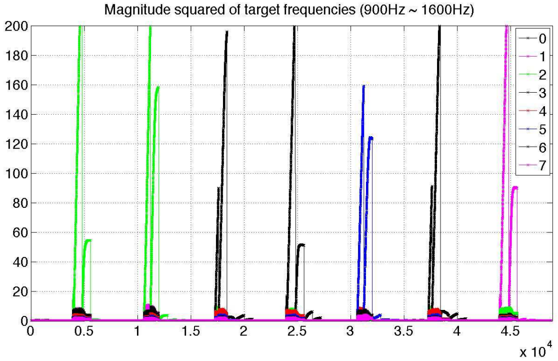

Figure 3: Magnitude squared of target frequencies 900Hz - 1600Hz. The decoding algorithm will process this to '2233531' (== 0x93759).

5.3 Visual Channel

MVSec leverages a strong OOB channel to transmit a short, temporary secret key to defend against the MitM attack. This OOB channel leverages the light bulb in a closed glove compartment to transmit messages, which will be detected by an ambient light sensor on the driver's phone. An Android phone is equipped with the sensor to measure the light intensity experienced by the phone. This sensor is generally used to detect light intensity for automatic brightness control and screen locking. We leverage Android's SensorManager class to implement the prototype. In our implementation, we fully implement the driver's smartphone, and simulate the car's glove box light source, by using another Android device, by varying the light intensity of the screen.

When the driver presses the start button on each device to initiate the protocol, the car will emit a sequence of light signals to the driver's smartphone. The signal is an encoding of a short temporary key (20 bits) as described in the protocol details in §4. Accounting for the low resolution of the ambient light sensor on the smartphones, the current prototype leverages four intensity levels to encode the corresponding bits: low, medium, high, and pause. Each level corresponds to the following lux values received by the receiver's ambient light sensor – 10 lx, 40 lx, 90 lx, and 160 lx.

[1] Due to the space limitation, we provide the details of the security analysis and AVISPA results in a technical report [9].

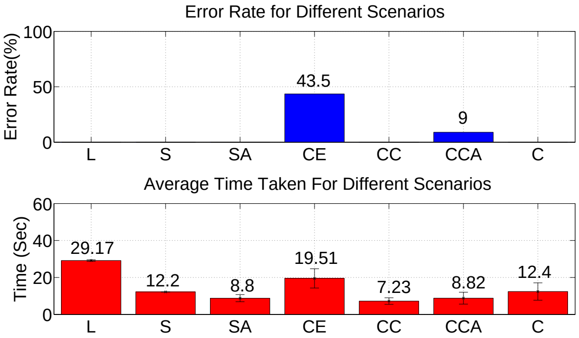

Figure 4: Error rate and time measurements of different study types. Attack scenarios are also included.

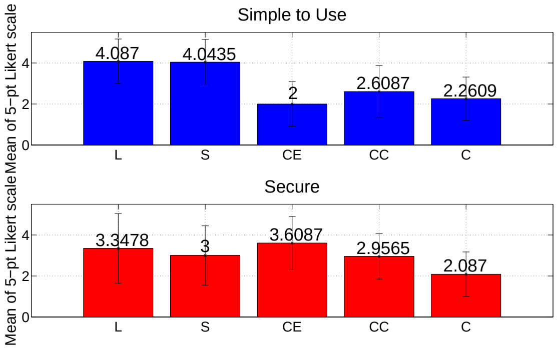

Figure 5: Post-test questionnaire results that rate user's perceptions for simplicity/security.

Due to the low sampling rate of the ambient sensor in the driver's phone, the car transmits one intensity level for every two seconds (one second for intensity value and another second for the pause bit), and takes a total of 26 seconds to transmit (20 bits encodes to 13 pulses). However, we envision that more responsive ambient sensors installed in newer phones will increase the speed.

MVSec uses the ambient light sensor as a proof-of-concept. However, we envision that using other sensors to read the light signal (e.g., camera) would increase the overall detection time and improve the performance.

6. EVALUATION

This section provides the evaluation of the usability, as well as the OOB channel detection accuracy. We present the results of the user study conducted by describing the participant profile, study process, and analysis of the results. We also evaluate how accurate the OOB channel is in terms of the detection accuracy. [2]

6.1 Usability Analysis

The main goal of this user study is to determine the usability of the MVSec. Specifically, we design our study to verify (1) whether MVSec reduces user errors as well as pairing timing, and (2) the user's perception of MVSec being more secure and simple to use compared to other solutions.

Demographics. We recruited 23 participants from different sources such as Craigslist and a university mailing lists (Varied participant pool in gender, age, and education background).

The participants' age range was 20–59; 13 are in twenties, 6 are in thirties, and 4 are in more than forties. These participants include 12 male and 11 female. Among 23 participants, 10 have undergraduate degree (e.g., master or doctorate degrees), 13 have college degree, and one participant has only high school diploma.

User Study Process. Participants are invited to the driver's seat in a car to perform user study. We present to them with two phones – one to simulate the car's control unit (P_{car}) and the other to be used as the driver's smartphone (P_{driver}). P_{car} is attached to the car's dashboard to simulate the vehicle's infotainment system. We designed both the *light (L)* and *sound (S)* MVSec scenarios to be tested for the user study. Although we fully implemented the working prototype for L scenario as mentioned in §5, we simulated L for the user study by asking the user to place P_{driver} into the glove compartment and explained to them that the light in the compartment will be emitting secret light signal to P_{driver}.

For comparison, we implemented three baseline cases that are currently used as Bluetooth pairing schemes in vehicles. The three cases are *choose-and-enter (CE)*, *compare-and-confirm (CC)*, and *copy (C)*. *CE* allows the user to choose a hard-to-guess number and enter it on both of the devices. *CC* allows the user to compare the numbers displayed on each of the devices. *C* allows the user to copy a displayed number on the car, and input it into his phone.

In addition to these five scenarios, we also added two attack scenarios – one for MVSec and the other for the baseline case. First, we present *sound attack (SA)*, an attack on *sound* by having an unintended device beep, when the participant is performing sound pairing scenario, and test if the participant is able to detect the beep from the unintended device and aborts the pairing process. Second, we present *compare-and-confirm attack (CCA)*, an attack on *CC* by presenting two numbers that are different by a digit, and test if the participant can determine the difference. To reduce bias between the subjects, we present the seven scenarios in random order for different participants.

Study Results. During the execution of the scenarios, we measure the following two outputs for comparison – *error rate* and *time*. For non-attack scenarios (i.e., L, S, CE, CC, and C), we claim that an error occurs when the participant performs tasks in an incorrect manner resulting in an unsuccessful pairing. For attack scenarios, (i.e., SA and CCA), an error occurs if the participant does not detect a problem, and continues the pairing procedure without aborting. Figure 4 depicts the comparison of the six scenarios with respect to error rate and timing.

The first graph in Figure 4 illustrates that the error rate is around 45% for CE, which is a significant percentage. This is because many participants chose easy-to-guess six digit number, when asked to come up with a six digit passkey. Because the security of this approach depends on the passkey to be unpredictable, this demonstrates a clear security problem. We also observe that for the attack scenario of *compare-and-confirm* (Scenario SA), about 10% of the participants mistakenly accepted different values displayed on the devices to be the same. However, we did not find any error caused by the participants when pairing via the L and S. More interestingly, during the attack scenario of S, all participants distinguished the beeps from the intended devices

[2] Due to space limitation, we present the detection accuracy of the audio channel of the prototype implementation discussed in §5.2 in our technical report [9].

as opposed to the unintended device, and pressed abort button as instructed.

The lower graph in Figure 4 depicts the average time taken for different scenarios. On average, L took around 29 seconds, which is the longest to complete, due to the low resolution of the ambient light sensor. CE followed L with around 20 seconds of average completion time. This is because the participants had to come up with a six digit passkey, and enter the number twice, once on each device. C and S took about the same time of around 12 seconds. The fastest average completion time was CC, because this scenario did not require the user to enter any numbers on the devices.

Upon completion of all seven scenarios, we asked the participants to rate the scenarios (excluding the attack scenarios) with a five point Likert scale for *simplicity* and *security* (scale from 1 to 5: 1 being the least simple/secure and 5 being the simplest/most secure). Figure 5 depicts the average of the Likert scale. It is interesting to note that both L and S have significantly higher average (both above average value 4) than the baseline cases for simplicity, despite the fact that L took the longest time to complete. It is also interesting to note that the user perception for security are relatively well distributed among different scenarios, fortifying the fact that the participants well represent average users without security expertise.

With the aforementioned results, we claim that MVSec provides a clear usability advantage over the baseline cases, which are used as industry standards in many of the vehicle-to-mobile pairing schemes. We find that MVSec simplifies user experience, while significantly reducing error rate.

7. DISCUSSION

We now discuss some of the relevant points that were not addressed in the above sections.

Alternative Pairing Methods. There are alternative pairing solutions that may seem to be valid at a first glance for performing a secure key agreement. However, we provide reasons for why they may not be adequate solutions. First, many cars are already equipped with built-in iPod jacks. While it is possible to perform secure key agreement using such cables, we find that not all existing cars today have such cables. We design MVSec to be deployed in all cars, including existing cars without such cables. Second, NFC may be used as an OOB channel to perform authentication. However, NFC suffers the same issue – not all cars are equipped with NFC chips today. To exacerbate this problem, many mobile devices today do not have NFC chips. In particular, iOS devices which have significant market share, ship without NFC chips. Hence, we find that NFC cannot meet our goal of deploying MVSec to all existing cars, while incurring minimal hardware cost.

Visual Channel. Recall that our solution leveraging visual channel was established by varying the light intensity in the glove compartment to emit signals to the phone inside the compartment. While current cars today only have a simple mechanical controller that turns on the light when the compartment door opens, we envision that the light source can be controlled by either installing a new ECU (Electronic Control Unit) or being controlled by existing ECU in the future. To support MVSec in existing cars, dealers can easily service existing cars to install such controllers.

Access Control Policy. MVSec employs an access control policy where the right to drive the vehicle equates to the right to pair a phone. In addition, the driver may delegate such rights to the passengers. However, there may be situations that such policy may not be sufficient. This is best exemplified when the driver leaves his car with valet parking or repair service center. If the glove compartment is unlocked, the valet or service personnel may pair their phones with the car. To resolve this issue, we envision MVSec to employ the following mechanism. MVSec may enforce the car to prompt the driver's phone for any additional pairing requests, so that the car would only proceed with the pairing process after the driver's authorization. (We assume that first phone to be paired does not require such authorization.)

8. CONCLUSION

Wireless device pairing is often vulnerable to MitM attacks. Thus, secure pairing between a vehicle and a phone is important for a successful industry deployment. The proposed protocols in this paper address solutions to protect against these attacks, while providing demonstrative identification to the human user. MVSec leverages readily available hardware to allow a car and a phone to perform secure key agreement without any pre-shared secret, and independent of a trusted third party, while still preserving usability.

9. REFERENCES

[1] W.F. Alliance. Wi-fi protected access: Strong, standards-based, interoperable security for today's wi-fi networks. *Retrieved March*, 1:2004, 2003.

[2] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuéllar, P. Drielsma, P. Heám, O. Kouchnarenko, J. Mantovani, et al. The avispa tool for the automated validation of internet security protocols and applications. In *Computer Aided Verification*, pages 135–165. Springer, 2005.

[3] Dirk Balfanz, Diana K. Smetters, Paul Stewart, and H. Chi Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *NDSS*, 2002.

[4] S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Proceedings of the IEEE Symposium on Security and Privacy*, 1992.

[5] Eric Cheng and Paul Hudak. Audio Processing and Sound Synthesis in Haskell. January 2009.

[6] Car Connectivity Consortium. Mirror Link. `http://www.mirrorlink.com/`.

[7] G. Goertzel. An algorithm for the evaluation of finite trigonometric series. *American Mathematical Monthly*, 65:34 – 35, 1958.

[8] Bluetooth Core Specification Working Group. Bluetooth simple pairing Whitepaper. Bluetooth SIG Whitepaper '06.

[9] Jun Han, Yue-Hsun Lin, Adrian Perrig, and Fan Bai. Mvsec: Secure and easy-to-use pairing of mobile devices with vehicles. In *CyLab Technical Report, May 2014, CMU-CyLab-14-006.*

[10] Cynthia Kuo, Jesse Walker, and Adrian Perrig. Low-cost manufacturing, usability, and security: An analysis of bluetooth simple pairing and wi-fi protected setup. In *USEC*, 2007.

[11] Sven Laur, N. Asokan, and Kaisa Nyberg. Efficient mutual data authentication using manuallyauthenticated strings. In *Cryptography and Network Security*, pages 90–107, 2006.

[12] Jonathan McCune, Adrian Perrig, and Michael Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2005.

[13] S. Pasini and S. Vaudenay. Sas-based authenticated key agreement. In *Theory and Practice of Public-Key Cryptography (PKC)*, 2006.

[14] Karen Scarfone and John Padgette. Guide to bluetooth security. *NIST Special Publication*, 800:121, 2008.

[15] Ersin Uzun, Kristiina Karvonen, and N. Asokan. Usability analysis of secure pairing methods. In *USEC*, 2007.

[16] Serge Vaudenay. Secure communications over insecure channels based on short authenticated strings. In *International Cryptology Conference (CRYPTO)*, 2005.

[17] Stefan Viehbock. Brute forcing Wi-Fi Protected Setup. When poor design meets poor implementation. `http://sviehb.files.wordpress.com/2011/12/viehboeck_wps.pdf`.