The 28th International Conference on Distributed Computing Systems Workshops

# Pairing Devices with Good Quality Output Interfaces

Nitesh Saxena and Jonathan Voris
Polytechnic University
nsaxena@duke.poly.edu, jvoris@isis.poly.edu

## Abstract

*The operation of achieving authenticated key agreement between two human-operated devices over a short range wireless communication channel, such as Bluetooth or Wi-Fi, is known as "pairing." The devices being paired are ad hoc in nature, i.e., they can not be assumed to have a prior context (such as pre-shared secrets) or a common trusted on- or off-line authority. However, the devices can generally be connected using auxiliary physical channel(s) (such as audio or visual) that can be authenticated by the user(s) of the devices. These authenticatable channels can be used to form a basis for pairing.*

*In this paper, we present the results of a user study of a technique used to pair two devices (such as two cell phones) which have good quality output interfaces in the form of a display, speaker, and/or vibration.*

***Keywords:*** *Authentication, Key Agreement, Device Pairing*

## 1 Introduction

Short range wireless communication, based on technologies such as Bluetooth and Wi-Fi, is becoming increasingly prevalent and promises to remain so in the future. With this surge in popularity comes an increase in security risks. Wireless communication channels are easy to eavesdrop upon and manipulate. Therefore, a fundamental security objective is to secure these channels of communication. In this paper we will use the term "pairing" to refer to the operation of bootstrapping secure communication between two devices connected by a short-range wireless channel. Examples of pairing operations performed during the course of daily life include connecting a Wi-Fi laptop and access point or a Bluetooth keyboard and desktop. Pairing would be easy to achieve if there existed a global infrastructure that enabled all personal devices to share an on- or off-line trusted third party, certification authority, PKI, or any pre-configured secrets. Such a global infrastructure is nearly impossible to come by in practice, however, thereby making pairing an interesting and challenging real-world research problem (the pairing problem has been at the forefront of various recent standardization activities, see [12]).

**Out-of-Band Channels.** A recent direction in pairing research is to use an auxiliary physically authenticatable channel, called an out-of-band (OOB) channel, which is governed by the human users who operate the devices. Examples of OOB channels include audio and visual channels. Unlike wireless channels, an adversary is assumed to be incapable of modifying messages transmitted on an OOB channel. It can eavesdrop on, delay, drop and replay them, however. A pairing scheme should therefore be secure against such an adversary. The usability of a pairing scheme based on OOB channels is clearly of the utmost importance. Since OOB channels typically have low bandwidth, the shorter the data that a pairing scheme needs to transmit over these channels, the better the scheme becomes in terms of usability.

Various pairing protocols have been proposed thus far. These protocols are generally based on bidirectional automated device-to-device (d2d) OOB channels. Such d2d channels require both devices to have transmitters and corresponding receivers. In settings where d2d channel(s) do not exist (i.e., when at least one device does not have a receiver) equivalent protocols can be based upon device-to-human (d2h) and human-to-device (h2d) channel(s) instead. Depending upon the protocol, only two d2h channels may be a sufficient replacement. This is the case when the user has to perform a very simple operation (such as "comparison") on the data received over these channels. Clearly, the usability of d2h and h2d channel establishment is even more critical than that of a d2d channel.

**Short Authenticated Strings.** Earlier pairing protocols required at least 80 bits of data to be transmitted over the OOB channels. The simplest protocol [1] involves the devices exchanging their public keys over the wireless channel and authenticating them by exchanging (at least 80-bit long) hashes corresponding to the public keys over the OOB channels. The more recent, so-called Short Authenticated Strings (SAS) based protocols [6][4] reduce the length of data transmitted over the OOB channels to approximately 15 bits (SAS-based authentication was first introduced by Vaudenay in [14]).

A number of pairing schemes that utilize various OOB channels have been proposed that are based on the protocols listed above. Recently, in [8] we proposed a new pair-

1545-0678/08 $25.00 © 2008 IEEE
DOI 10.1109/ICDCS.Workshops.2008.87

ing scheme that is universally applicable to any two devices, with a focus on devices that lack good quality output interfaces. This scheme can be based on any existing SAS protocol and does not require devices to have good transmitters or any receivers, e.g., just a pair of LEDs is sufficient. The scheme involves users comparing very simple audio-visual patterns, such as "beeping" (using a basic speaker) and "blinking" (using LEDs), transmitted as simultaneous streams which form two synchronized d2h channels.

**Contributions.** In this paper we focus on pairing devices, such as cell phones, that have good quality output interfaces in the form of a display, speaker, and vibration mechanism. We apply the same basic pairing concept that was used in [8]. In this paper, however, we take advantage of the higher quality output interfaces which are often available on devices of this kind. The analysis of pairing solutions that use different methods of output is a worthwhile goal unto itself, as many current solutions are of limited use to disabled individuals. Besides this, we investigate whether or not the usability and efficiency of the scheme presented in [8] can be improved if the pairing devices have better, and in some cases, multiple, output interfaces. We extend the scheme of [8] to create three new OOB output combinations: Flash-Flash, Vibrate-Vibrate, and All-All. We also compare these to a simple scheme (referred to as Num-Num) that requires users to compare two 4-digit numbers displayed on devices' screens.

The results of our user tests of these schemes indicate that none of them are superior to the others in all respects. However, our tests show that All-All is the safest in that it has very low false negatives, whereas Vibrate-Vibrate turns out to be the most user friendly.

## 2 Communication and Security Model

The pairing protocols used in this paper are based upon the following communication and adversarial model [14]. The devices to be paired are connected via two types of channels: (1) a short-range, high-bandwidth, bidirectional wireless channel and (2) an auxiliary, low-bandwidth, physical OOB channel(s). An adversary attacking the pairing protocol is assumed to have full control on the wireless channel; namely, he or she can eavesdrop on, delay, drop, replay and modify messages. On the OOB channel, the adversary can eavesdrop on, delay, drop, replay, and re-order messages. It can not modify them, however. In other words, the OOB channel is assumed to be authenticated.

To date, two three-round pairing protocols based on short authenticated strings (SAS) have been proposed [6][4]. In a communication setting involving two users restricted to running three protocol instances, these SAS protocols need to transmit only $k$ $(= 15)$ bits of data over an OOB channel. As long as the cryptographic primitives used in these protocols are secure, an adversary attacking them can not win with a probability significantly higher than $2^{-k}$ $(= 2^{-15})$. This provides security equivalent to that provided by 5-digit PIN-based ATM authentication [14].

## 3 Pairing Using Synchronized Outputs

In this section we describe the design and implementation of our pairing schemes based on the Flash-Flash, Vibrate-Vibrate and All-All OOB output combinations. We also compare these combinations with the simple comparison scheme Num-Num.

**Our Goals.** Our objective was to develop pairing schemes that leverage the decent quality output interfaces found on most ubiquitous personal devices such as mobile phones, headsets, and remote controls. This is unlike the motivation for the schemes of [8], which focused on devices that lack such methods of output. Note that while these devices typically feature multiple methods of output, their input interfaces vary and may be limited. Thus it is difficult to design d2d channels to pair this class of devices in a general way. To keep our pairing technique applicable to as many devices as possible, we do not discuss the use of d2d channels. For more on pairing with d2d channels see [1], [5], and [9].

The output channels that we utilize in this paper's schemes are a "flashing" screen, "vibration," and "beeping" using a speaker. What we call Flash-Flash is implemented via the flashing of backlit LCD screen, while Vibrate-Vibrate uses vibration functionality commonly associated with cell phones. We wanted to determine how to make use of the various output interfaces available on these devices in a way that improved the pairing procedure. We looked to enhance the pairing experience by making the process as short and simple as possible, thus placing a minimal burden on device users. In particular, since devices of this kind usually feature more than one method of output, we desired to test whether the use of multiple simultaneous output channels made the pairing process any easier for users. New types of OOB channels also improve the pairing experience by making pairing suitable for a greater range of users. Devices restricted to a visual output interface for pairing are not usable by the visually impaired, for instance.

**The Design.** To achieve these goals, we implemented All-All using the aforementioned flashing display and vibration functionality as well as a beep in the form of a brief alert noise or "ringtone." Since the results of [8] indicated that human users generally do not prefer "asymmetry" in OOB output, such as when two devices use different output interfaces, we decided not to proceed with combinations that used a different output channel on each device, such as Flash-Vibrate, etc. Moreover, since the pairing combination involving two similar audio-based output channels from [8], Beep-Beep, was error-prone, no further experimentation was conducted with that combina-

tion. We did perform user testing with an additional pairing scheme, Num-Num, however. Unlike the other schemes, Num-Num does not make use of any synchronized patterns of OOB data. Instead, it requires users to visually compare the two 15-bit SAS messages after they have been encoded as two 4-digit numbers and displayed on the screen of each of the pairing devices. Since Num-Num appears to be the most basic and simple pairing technique possible, we wanted to see how it compared with the three new OOB output combinations, Flash-Flash, Vibrate-Vibrate, and All-All.

A pairing scheme, in its entirety, consists of three phases: (1) the device discovery phase, wherein the devices exchange their identifiers over the wireless channel prior to communicating, (2) the pairing protocol execution phase, wherein the devices execute the desired pairing protocol over the wireless channel, and (3) the authentication phase, where the devices authenticate the messages exchanged during the previous phase using OOB channels. For the sake of our experimentation, we skipped the first two phases and concentrated on the third phase. We did this because our main goal was to test the feasibility of the way we intended to implement the OOB channels,

Let us assume that we want to pair two devices, $A$ and $B$. Assuming that $A$ and $B$ have already performed the device discovery and protocol execution phases over the wireless channel, the pairing task is then reduced to $A$ and $B$ encoding their respective 15 bits of SAS data, $SAS_A$ and $SAS_B$, into vibrating, beeping, flashing, or some combination of these outputs and then transmitting this encoded data in a synchronized fashion for the user to compare (except for the Num-Num combination, which clearly does not require any synchronization). This encoding should enable the user to easily identify both "good" cases, i.e., when $SAS_A = SAS_B$, and "bad" cases where $SAS_A \neq SAS_B$.

In [8], synchronization was achieved by having one device send a synchronization signal $S$ to the other device over the wireless channel. However, this synchronization signal can get or be delayed, resulting in users being fooled into accepting non-matching SAS data (for example, the strings "010010" and "100100" will appear to be equal to the user if the synchronization signal is delayed by one bit). In [8] this issue was dealt with by using an END marker to indicate the completion of each SAS. For the pairing schemes in this paper we opted to use a different synchronization technique. Since these schemes are targeted at devices that are hand-held and/or easy to manipulate, we assumed that users would be able to simultaneously press a button to activate the pairing process on two devices. This is a reasonable assumption for a wide range of cell phone-like devices. This approach prevents potential synchronization delays by placing the timing of the encoded SAS exchange in the hands of the user while also avoiding the added complexity introduced by the use of $S$ and an END marker.

**Encoding Method.** For all of our OOB output combinations, a '1' bit in the SAS data is signaled by activating an output interface for a given "signal interval," while a '0' bit is represented by disabling an output for the same interval. Every bit signal is followed by a brief "sleep interval" of no output. As an example, on a '1' bit the pairing devices would vibrate for 210 ms and then stay still for 490 ms, while on a '0' bit the device would remain stationary for the entire 700 ms. This is included to facilitate "human-comparison" by allowing users to differentiate two separate, consecutive bit signals. The time required to compare two SAS strings is inversely proportional to the duration of the signal and sleep intervals. The optimal value for these intervals needs to be determined through experimentation. The best interval duration is a careful balance between granting users enough time to comfortably compare the encoded SAS data and making the comparison process as brief as possible.

The output interface(s) used for the encoding depends on the OOB combination in use. For Flash-Flash, a display backlight is brightened and darkened to encode the SAS data. Vibrate-Vibrate utilized the vibration function commonly used for feedback in cell phones, headsets, and video game controllers. The All-All scheme was designed to test whether users find multiple simultaneous forms of OOB output helpful or distracting. As such, this method makes use of the aforementioned "flashing" and "vibrating" outputs as well as an audio, or beeping," output. This type of encoding is "all-or-nothing" in the sense that a device either outputs all three types of feedback or none at all.

**Implementation.** We used two Nokia 6030b mobile phones to conduct our experiments. This phone model was selected for testing because they are affordable entry-level models that have been commercially available for a few years. As such, their features are representative of those one would expect to find on today's average personal mobile device. These phones have several good quality output interfaces, specifically a vibration feature, a speaker, and a 16-bit color, 128 by 128 pixel display. The Nokia 6030b runs the Nokia operating system and supports version 2.0 of the Mobile Information Device Profile (MIDP) specification and version 1.1 of the Connected Limited Device Configuration (CLDC) framework, which are both part of the Java Platform, Micro Edition, or J2ME. To utilize these APIs we wrote our test programs in the Java programming language using the Java Wireless Toolkit version 2.5.2 for CLDC. Because we were only working with the authentication phase of the pairing scheme and not the device discovery or pairing protocol execution phase and did not make use of a wireless channel for synchronization, no actual wireless connection between the two mobile devices was necessary for our tests.

## 4 Usability Testing

Next, we present an experimental study of the usability of our new pairing setups. These schemes were tested with a total of 40 subjects. All of our testers were college students who were familiar with mobile phones but not particularly proficient with the technology. Each tester was given a short verbal summary of our secure device pairing schemes and their potential applications. We explained to the volunteers the experimental setup of the two devices and what they were expected to do while working with the four output combinations.

The primary objective of our user tests was to determine which of the output combinations enabled them to most easily identify SAS signal matches and mismatches. Put differently, we wanted to determine which encoding caused users to commit the least amount of *safe errors* (a false positive, that is, identifying a match as a mismatch) and *fatal errors* (a false negative, that is, identifying a mismatch as a match) [13]. The secondary goal of our tests was to establish an optimal timing interval for the type of output that users were most comfortable with.

### 4.1 Test Setup

The experiments for the four output combinations were conducted in a graduate research lab of our university. The test cases presented to the volunteers were designed to test for SAS matches and mismatches. The strings utilized in these test cases were randomly generated, but fixed from subject to subject to prevent some volunteers from receiving strings that were easier to identify than others (most users would not have a problem differentiating between the encoding of '0000,' and '1111,' while the more subtle difference between the strings '1010' and '0101' might be harder to notice). These strings were presented to the test subjects in a random order. This was done to minimize the effects of learning and fatigue on the test results. In other words, we wanted to prevent users from anticipating future test cases based on previous ones or losing motivation to pay proper attention during the pairing procedure.

During our preliminary tests we observed that users had an easier time noticing mismatches that occur towards the front of a SAS if the string was prepended with a few non-data "padding" bits. This observation is corroborated by the results of [8]. These initial bits give users an opportunity to focus their attention on the upcoming SAS bits. We used 3 bits of padding, which combined with the 15 bits of SAS data to produce 18-bit long test case strings. Prior to each test case, the test administrator would configure the test parameters on both phones, such as which SAS and what timing interval to use. Volunteer users then initiated the test process by pressing a button on each phone simultaneously. After a very brief (100 ms) delay both devices started signaling their particular SAS test string via vibrating, flashing, or vibrating, flashing and beeping in accordance with the OOB combination in use for that particular test case. For the Num-Num output combination, users were not required to activate the test on both devices simultaneously. They were instead asked to press a button on each device that would cause a 4 digit decimal encoding of the SAS value to appear on the phone's display. Test subjects were then asked to compare the values displayed on the two devices.

### 4.2 Vibrate-Vibrate Tests

Since vibration was a novel output interface that had not yet been tested in previous research, we decided to test the viability of Vibrate-Vibrate before comparing it to other types of output. We set out to find the best timing interval for this pairing scheme with respect to user comfortability and the overall runtime of the pairing procedure. We experimented with a number of timing intervals ranging between 300 and 800 ms. Each overall interval consisted of approximately 30% signal interval and 70% sleep interval. For example, an overall timing interval of 300 ms corresponded to 80 ms of signal interval and 220 ms of sleep interval. Each of the 20 volunteers for these tests performed 20 Vibrate-Vibrate pairing test cases for a total of 400 test runs (4 users also participated in 35 preliminary test cases, mentioned above, for a grand total of 435 tests). The outcome of our tests of Vibrate-Vibrate with different intervals are depicted in Table 1.

Our test results indicate that users commit errors when Vibrate-Vibrate is used with an interval of 300 to 650 ms. The fatal error rates for this scheme at these speeds range from 2% for a 500 ms interval to approximately 19% for a 550 ms interval. It is worth noting, however, that these error rates are comparable to those encountered when using the output combinations presented in [8]. The most promising results occur at the 700 to 800 ms interval range that follows, where the entire comparison process takes between 13 and 15 seconds to complete. At this level no errors were committed at all. Thus, at this interval range users can easily and comfortably detect all SAS errors. This shows that as the sleep interval duration rises, users feel increasingly comfortable performing comparisons using Vibrate-Vibrate.

### 4.3 Comparison Tests

Having confirmed the usability of Vibrate-Vibrate, we desired to see how the scheme stood up to other methods of output. Furthermore, since we were working with devices with more than one output interface, we also wanted to compare single output combinations like Vibrate-Vibrate and Flash-Flash to a multiple output combination, All-All. This comparison was designed to analyze whether multiple simultaneous output channels made the pairing process easier or more difficult for users.

| Interval (ms) | Safe Error Rate (%) | Fatal Error Rate (%) |
|---|---|---|
| 300 | 16.667 | 5.556 |
| 400 | 6.122 | 4.082 |
| 500 | 10.000 | 2.000 |
| 550 | 6.250 | 18.750 |
| 600 | 4.167 | 8.333 |
| 650 | 4.348 | 8.696 |
| 700 | 0.000 | 0.000 |
| 750 | 0.000 | 0.000 |
| 800 | 0.000 | 0.000 |

**Table 1. Outcome of the Vibrate-Vibrate tests with 20 users**

Based on our initial tests, we found the optimal time interval values for Flash-Flash and All-All to be 1000 ms and 700 ms, respectively. We then performed user tests to compare Vibrate-Vibrate (with a 700 ms interval), Flash-Flash (with a 1000 ms interval), All-All (with a 700 ms interval), and Num-Num (which, as a non-synchronized scheme, intervals do not apply to). The 20 subjects who volunteered for this round of testing performed a total of 60 test cases for each of the four combinations. The results of these comparison tests are depicted in Tables 2 and 3. The timing data is averaged over all test runs and takes user reaction time into account.

Unlike our Vibrate-Vibrate specific tests, this time users did commit some errors when using each combination at these interval levels. Out of the four schemes we tested, users committed the least amount of safe errors – in fact none – when using the Num-Num combination. Num-Num also turned out be the fastest since it involved direct visual comparison instead of synchronized device output. Fatal errors were the lowest, about 1.67%, with All-All. Users committed a number of both safe and fatal errors with Vibrate-Vibrate and Flash-Flash, with Vibrate-Vibrate having less of both. Upon test completion users were asked to give a qualitative ranking of the output combinations in terms of ease of use. About 54% of the test subjects found Vibrate-Vibrate to be the best, while about 36% picked Num-Num and only about 9% selected All-All. None of the subjects rated Flash-Flash as the best.

### 4.4 Interpretation

These test results and user comments indicate that **none** of the schemes can be considered the overall best. All-All and Num-Num turned out to be more or less complementary to each other in that All-All has the lowest fatal error rate while Num-Num has the lowest safe error rate. This means that providing users with three simultaneous outputs as done with All-All does make users more alert to the presence of non-matching instances. Unfortunately, it can also confuse users, causing them to mistake a matching instance for a mismatch. Analogously, Num-Num made it quite easy for users to detect matching instances. It also made them too inattentive at times, however, which cased test subjects to miss some non-matching instances. Flash-Flash and Vibrate-Vibrate showed intermediary amounts of error rates, with users missing a few matching as well as a few non-matching instances. While there were some exceptions, users generally stated that they found the visual-only output of Flash-Flash difficult to focus on and the multiple outputs of All-All to be distracting. On the other hand, users found that the straightforward tactile feedback of Vibrate-Vibrate demanded less attention and was easier to keep track of than the other combinations.

| Combination | Average Timing (seconds) | Safe Error Rate (%) | Fatal Error Rate (%) |
|---|---|---|---|
| Vibrate-Vibrate | 14.3 | 3.333 | 5.000 |
| Flash-Flash | 18.0 | 8.333 | 10.000 |
| All-All | 15.2 | 10.000 | 1.667 |
| Num-Num | 2.1 | 0.000 | 9.091 |

**Table 2. Outcome of the four output combination tests with 20 users**

| Combination | Ranked #1 (%) | Ranked #2 (%) | Ranked #3 (%) | Ranked #4 (%) |
|---|---|---|---|---|
| Vibrate-Vibrate | 54.55 | 9.09 | 36.36 | 0.00 |
| Flash-Flash | 0.00 | 45.45 | 18.18 | 36.36 |
| All-All | 9.09 | 27.27 | 18.18 | 45.45 |
| Num-Num | 36.36 | 18.18 | 27.27 | 18.18 |

**Table 3. Responses of 20 users when asked to compare the four output combinations**

## 5 Related Work

There exists a significant amount of prior work on the general topic of pairing. Due to space constraints we only summarize it here. For more detailed descriptions, refer to the related work section of [8].

In their seminal work, Stajano, et al. [11] proposed the establishment of a shared secret between two devices using a link created through a physical contact (such as an electric cable). Balfanz, et al. [1] extended this approach through the use of infrared as a d2d channel; the devices exchange their public keys over the wireless channel, then exchange (at least 80-bit long) hashes of their respective public keys over the infrared channel.

The Snowflake mechanism by Levienet et al. [2] and the Random Arts visual hash by Perrig et al. [7] can be used to pair devices based on the comparison of random images.

McCune et al. proposed the "Seeing-is-Believing" (SiB) scheme [5]. SiB involves establishing two unidirectional

visual d2d channels; one device encodes the data into a two-dimensional barcode and the other device reads it using a still camera.

Goodrich, et al. [3] proposed a pairing scheme based on "Mad Lib" sentences that is also built upon the protocol of Balfanz et al. The main idea of their procedure is to establish a d2h channel by encoding the pairing data into English sentences, which users can then easily compare.

As an improvement to SiB, Saxena et al. [9] proposed a new scheme based on a visual OOB channel. The scheme uses one of the SAS protocols [4] and is aimed at pairing two devices of which only one has a relevant receiver.

A very recent proposal, [10], focuses on pairing two devices with the help of "button presses" by the user. This scheme is based upon a protocol that first performs an unauthenticated Diffie-Hellman key agreement, then authenticates the established key using a short password. Such a short password can be agreed upon between the two devices via three protocol variants that make use of button presses.

Uzun et al. [13] carry out a comparative usability study of simple pairing schemes. They consider pairing scenarios where devices are capable of displaying 4 digits of SAS data.

## 6 Conclusion

Based on the results presented in section 4, we can draw the following conclusions regarding the applicability of the Vibrate-Vibrate, Flash-Flash, All-All and Num-Num OOB output combinations to pairing two devices which have decent quality output interfaces. Clearly, no single scheme can be said to be the best in all regards. In practice, both safe as well as fatal errors will most likely be encountered while using any of these combinations. Since Flash-Flash suffered from both types of errors and no users preferred it, we believe that the Flash-Flash combination is not a good pairing option. This leaves Vibrate-Vibrate, All-All, and Num-Num remaining. We noticed that All-All and Num-Num are complementary to each other in terms of errors. All-All had the lowest occurrence of fatal errors while Num-Num had the least safe errors (none). This means that All-All is the *safest* of the combinations we tested as it is the one that makes it the easiest for users to detect non-matching instances (potentially arising due to attacks). Conversely, Num-Num makes it easier for users to detect matching instances. However, it also tends to make users too inobservant which causes them to accept some non-matching instances as well. Based on our tests of the 4-digit Num-Num method, we feel that it would be risky to use as is. Perhaps comparison of longer numbers would help reduce the amount of fatal errors with respect to Num-Num [13], as these may force users to be more attentive.

Although it showed slightly higher safe error rates than Num-Num and sightly higher fatal error rates than All-All, Vibrate-Vibrate was the combination that most users preferred. Our results undoubtedly indicate that Vibrate-Vibrate is the *most usable*. We feel that since users liked this scheme so much it has the most potential for improved security and may yield much lower error rates in practice. Of course, Vibrate-Vibrate is only applicable to pairing two devices with vibration capabilities

In closing, the importance of making different OOB channels available for the disabled must be noted. The Vibrate-Vibrate combination is an excellent pairing choice for bootstrapping communication between devices deigned for the visually impaired, as visual techniques are not applicable to such devices. Audio methods are also of limited use in this scenario due to their obtrusiveness in terms of noise and the difficulty users have in determining the actual source of a sound. This was demonstrated by Beep-Beep in [8], which was shown to be highly error-prone for this reason. In general, devices with a greater range of pairing output options will be accessible to a wider range of people.

## References


[1] D. Balfanz, D. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *NDSS*, 2002.

[2] I. Goldberg. Visual Key Fingerprint Code, 1996. Available at http://www.cs.berkeley.edu/iang/visprint.c.

[3] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and Clear: Human-Verifiable Authentication Based on Audio. In *ICDCS*, 2006.

[4] S. Laur, N. Asokan, and K. Nyberg. Efficient mutual data authentication based on short authenticated strings. IACR Cryptology ePrint Archive: Report 2005/424, 2005.

[5] J. M. McCune, A. Perrig, and M. K. Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *IEEE Symposium on Security and Privacy*, 2005.

[6] S. Pasini and S. Vaudenay. SAS-Based Authenticated Key Agreement. In *PKC*, 2006.

[7] A. Perrig and D. Song. Hash visualization: a new technique to improve real-world security. In *CrypTEC*, 1999.

[8] R. Prasad and N. Saxena. Efficient device pairing using "human-comparable" synchronized audiovisual patterns. In *ACNS*, to appear June 2008.

[9] N. Saxena, J.-E. Ekberg, K. Kostiainen, and N. Asokan. Secure device pairing based on a visual channel. In *IEEE Symposium on Security and Privacy (ISP'06), short paper*, 2006.

[10] C. Soriente, G. Tsudik, and E. Uzun. BEDA: Button-Enabled Device Association. In *IWSSI*, 2007.

[11] F. Stajano and R. J. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Security Protocols Workshop*, 1999.

[12] J. Suomalainen, J. Valkonen, and N. Asokan. Security associations in personal networks: A comparative analysis. In *ESAS*, 2007.

[13] E. Uzun, K. Karvonen, and N. Asokan. Usability analysis of secure pairing methods. In *USEC*, 2007.

[14] S. Vaudenay. Secure communications over insecure channels based on short authenticated strings. In *CRYPTO*, 2005.