Secure Device Pairing Based on a Visual Channel: Design and Usability Study

Nitesh Saxena
Polytechnic Institute of NYU, USA
nsaxena@poly.edu
Jan-Erik Ekberg, Kari Kostiainen, N. Asokan
Nokia Research Center, Helsinki, Finland
{jan-erik.ekberg, kari.ti.kostiainen, n.asokan}@nokia.com

***Abstract*— "Pairing" is the establishment of authenticated key agreement between two devices over a wireless channel. Such devices are ad hoc in nature as they lack any common pre-shared secrets or trusted authority. Fortunately, these devices can be connected via auxiliary physical (audio, visual, tactile) channels which can be authenticated by human users. They can therefore be used to form the basis of a pairing operation.**

Recently proposed pairing protocols and methods are based upon bidirectional physical channels. However, various pairing scenarios are asymmetric in nature, i.e., only a unidirectional physical channel exists between two devices (such as between a cell phone and an access point). In this paper, we show how strong mutual authentication can be achieved even with a unidirectional visual channel, where prior methods could provide only a weaker property termed as *presence*. This could help reduce the execution time and improve usability of prior pairing methods. In addition, by adopting recently proposed improved pairing protocols, we propose how visual channel authentication can be used even on devices that have very limited displaying capabilities, all the way down to a device whose display consists of a cheap single light-source, such as an LED. We present the results of a preliminary usability study evaluating our proposed method.

I. Introduction

The popularity of short-range wireless technologies like Bluetooth and Wireless Local Area Networking (WLAN) based on the IEEE 802.11 family of protocols is experiencing enormous growth. Newer technologies like Wireless Universal Serial Bus[1] are around the corner and promise to be as popular. This rise in popularity implies that an ever increasing proportion of the users of devices supporting short-range wireless communication are not technically savvy. Such users need very simple and intuitive methods for setting up their devices. Since wireless communication is easier to eavesdrop on and easier to manipulate, a common set up task is to initialize secure communication. In this paper, we will use the term *pairing* to refer to this operation.[2]

Consequently, both security researchers and practitioners have been looking for intuitive techniques for ordinary users to be able to securely pair their devices. Although the primary impetus comes from the need to secure short-range wireless communication, the issue of intuitive security initialization is more generally applicable whenever ordinary users need to set up secure communication without the help of expert administrators or trusted third parties.

The pairing problem is to enable two devices, which share no prior context with each other, to agree upon a security association that they can use to protect their subsequent communication. Secure pairing must be resistant to a man-in-the-middle adversary who tries to impersonate one or both of these devices in the process. The adversary is assumed to be capable of listening to or modifying messages on the communication channel between the devices. One approach to secure pairing is to use an additional physically authenticatable channel, called an out-of-band (OOB) channel which is governed by humans, i.e., by the users operating these devices. The adversary is assumed to be incapable of modifying messages on the OOB channel, although it can listen to them. (It is important to note that this approach only requires the OOB channel to be authenticated but not secret, in contrast to the standard Bluetooth pairing based on "user-selected" secret PINs).

There has been a significant amount of prior work on building secure pairing protocols using OOB channels (we review these in Section VIII of the paper). They consider different types of OOB channels including physical connections, infrared, etc. Most closely related to our proposal is the "Seeing-is-Believing" (SiB) system [15], where the OOB channel is implemented as a visual channel. The SiB visual channel consists of a two-dimensional barcode of [20], displayed by (or affixed to) a device A, that represents security-relevant information unique to A. A user can point another camera-equipped device B at the barcode so that B can read the barcode visually, and use this information to set up an authenticated channel to A. If both devices are camera-equipped, they can mutually authenticate each other. "Authentication" in this case is based on demonstrative identification [1] rather than with respect to a claimed name.

SiB and other prior pairing methods are all based upon bidirectional physical channels. However, various pairing scenarios are asymmetric in nature, i.e., only a unidirectional physical channel exists between two devices (such as between a cell phone and an access point or between a desktop and a keyboard).

A preliminary version of this paper appeared in [22]

[1] http://www.usb.org/developer/wusb

[2] The term *pairing* was introduced in the context of Bluetooth devices. Other roughly synonymous terms include "bonding," and "imprinting".

Our Contributions: In this paper, we focus on asymmetric pairing scenarios and propose novel approaches to pairing for such scenarios. Our contributions are as follows:

1) We show how strong mutual authentication can be achieved using just a unidirectional visual channel. This results in two improvements:
   a) strong authentication becomes possible in situations where prior methods (such as SiB) could only achieve a weaker property termed as "presence".
   b) execution time for mutual authentication decreases significantly and usability improves.
2) By adopting recently proposed improved protocols [13], [17], we show how visual channel authentication can be used even on (interface-constrained) devices that have very limited displaying capabilities, all the way down to a device whose display consists of a cheap single flashing light-source, such as a single light-emitting diode (LED). The proposed pairing method is most suitable for pairing a camera phone with an access point or a headset.
3) We perform a usability study evaluating our camera-based pairing method. Our results indicate that the proposed approach is reasonably efficient and quite user-friendly. Since usability is an important criterion for the adoption of a pairing method in practice, our usability study represents a significant extension to the preliminary version of this paper [22]. Our study provides new insights on the efficiency and acceptability of the method, and also points out that the efficiency can improve as users become more and more familiar with the method. We note that our pairing speed of around 40 seconds compares favorably to the speed of another pairing method [24], which is the leading alternative for pairing scenarios involving a device with constrained interfaces (such as an access point or a headset).

We remark that our new pairing protocols that require only a unidirectional OOB channel also extend well to other OOB channels (such as audio and tactile). Our approach has been employed as a foundation for other pairing proposals including the two variants presented in [28] and [25]. In addition, our proposal can also be used to improve the usability and security of other pairing methods, including [10] and [24]. We will discuss these applications of our approach to other pairing methods in Section VIII.

The rest of the paper is organized as follows. First, we start with a brief description of SiB in Section II. In Section III we describe an alternative protocol that improves the presence guarantee provided by SiB to full-fledged mutual authentication. Then, in Section IV, we show how visual channel authentication can be done even in highly constrained environments. We discuss the applicability and relevance of our improvements and extensions in Section VII. Finally, in Section VIII, we discuss various different pairing methods and compare them with the approach we propose in this paper.

II. Seeing-is-Believing

Several researchers have proposed the idea of encoding service or device discovery information in the form of barcodes so that they can be read using camera phones [20], [4], [30], [14]. The idea of encoding cryptographic material into barcodes was first proposed by Hanna [11] as well as Gehrmann, et al. [7], both of which also mention the use of asymmetric key cryptography in this context. These proposals, however, assume the barcode-enabled OOB channel to be both authenticated and secret. The SiB paper [15] by McCune et al. was the first research paper that proposes that the information encoded in the barcode be only authenticated (and not secret).

In SiB, a device A can authenticate to a device B, if B is equipped with a camera. A's commitment to its public key (such as a hash) is encoded in the form of a two-dimensional barcode of [20]. A typical barcode has dimensions approximately 2.5x2.5 cm^2 to allow recognition from a reasonable distance, and consists of a total of 83-bits of information (68-bits of data and 15-bits for forward error correction). If A has a display, the public key can be ephemeral, and the barcode is shown on the display. Otherwise, A's public key needs to be permanent and the barcode is put on a printed label affixed to the housing of A. Authentication is done by the user pointing B's camera at A's barcode. The basic unidirectional authentication process – using which A is authenticated to B – is depicted in Figure 1.

1) A calculates h_A as $h(K_A)$
   $A \longrightarrow B$ (visual channel): h_A
2) $A \longrightarrow B$ (insecure channel): K_A
   B calculates h' as $h(K_A)$ using the K_A received.
   If h' does not match the h_A received in Step 1, B aborts.

Fig. 1. SiB unidirectional authentication protocol (A is authenticated to B)

K_A is A's public key. $h()$ is a cryptographic hash function, which is resistant to second pre-image finding. K_A can be long-lived, in which case the output of $h()$ must be sufficiently large, e.g., at least 80-bits. If K_A is ephemeral, the output of $h()$ can be smaller, e.g. 48 bits [8]. SiB could accommodate 68 bits of hash into a single two-dimensional barcode, but requires a good quality display due to the typical size of the barcode[3].

Mutual authentication – using which A is authenticated to B and B is authenticated to A – requires the protocol of Figure 1 being run in each direction. This has two implications for SiB.

- First, mutual authentication is possible only if **both** devices are equipped with cameras. McCune, et al. state (Section 7 of [15])

  > A display-only device ...is unable to strongly authenticate other devices using SiB ...[because it] cannot "see" them.

  Let us say that device B has a camera but A does not have a camera. In this scenario, SiB can be used for authenticating A to B because B can capture the barcode displayed on A. However, A can not authenticate B as it is not possible for A to capture B's barcode. McCune, et al. suggest that a camera-less device (A) can only

[3]SiB can encode the data into several barcodes displayed in sequence.

achieve a weaker property known as *presence*. This is achieved by A including a secret key K in the barcode. The camera-equipped device B that reads the barcode can use K to compute message authentication code (MAC) over the message it sends to A (over the wireless radio channel). If the MAC is correct, A can conclude that it was sent by some device that was able to "see" its barcode, and thus was "present". Presence is a weaker security notion than authentication because A has no means of knowing if B is really the device that the user of A intended to communicate with. In other words, any other (adversarial) device in the close proximity of A that can capture A's barcode can satisfy the requirement of presence. In summary, if B has a camera, but A does not have a camera, SiB can be used to authenticate A to B, but can only guarantee to A the presence of B.

We summarize the types of authentication achievable using SiB for given combinations of device types in Table II.

- Second, in order to run the protocol in each direction, the roles of the devices have to be switched so that first A's camera can scan B's display and then B's camera can scan A's display. Such switching of devices by users not only increases the execution time of the SiB process but also decreases usability. McCune, et al. report that the average SiB execution time in their user trials was 8 seconds, even though time required to recognize a barcode is just about one second [20].

These implications limit the applicability of SiB in various practical settings. Many devices cannot have either cameras or high quality displays for different reasons. Commoditized devices like WLAN access points are extremely cost-sensitive and the likelihood of adding new hardware for the purpose of authentication is very small. Devices like Bluetooth headsets are typically too small to have displays or even to affix static barcode stickers.

To summarize, we identify the following drawbacks with the basic SiB scheme:

1) Mutual authentication is not possible unless both devices are equipped with cameras.
2) The overall execution time for mutual authentication is high, which impacts usability.
3) Applicability of SiB is limited in situations where one device has limited capabilities (e.g., small size, no camera, limited or no display at all).

In the rest of this paper, we describe how we can address each of these drawbacks.

III. Seeing Better: Upgrading Presence to Authentication

In this section, we address the issue of mutual authentication. Recall that we identified two shortcomings of SiB in this respect. First, SiB can provide mutual authentication only if *both* devices are camera-equipped. Second, the processing time for mutual authentication is high.

We observe that both of these drawbacks stem from the fact that mutual authentication is done as two separate unidirectional authentication steps. Therefore, we propose to solve both problems by performing mutual authentication in a single step by having each of A and B compute a *common* checksum on public data, and compare their results via a unidirectional transfer using the visual channel. Let us call this protocol VIC, for "Visual authentication based on Integrity Checking." (See Figure 3.)

Because VIC needs only a unidirectional visual channel, it is now possible to achieve mutual authentication in the cases where SiB could only achieve presence. In addition, the execution time for mutual authentication and the user effort will be less since no device role switching is required anymore. Thus, VIC addresses the first two drawbacks of SiB identified in Section II.

In Table 4, we summarize the types of authentication achievable using VIC for given combinations of device types. Notice that since the checksum is different for each instance of VIC, at least one device must have a display and that the static barcode labels cannot be used with VIC.

Security Analysis of VIC: Now we argue the security of VIC. To do so, we first preview the adversarial model for the authentication protocols based on OOB channels, as described in [29]. The devices being paired are connected via two types of channels: (1) a short-range high-bandwidth wireless channel, and (2) auxiliary low-bandwidth physical OOB channel. An adversary attacking the protocol is assumed to have full control on the wireless channel, namely, it can eavesdrop, and modify messages. On the OOB channel, on the other hand, the adversary can eavesdrop, but can not modify the messages. In other words, the OOB channel is assumed to be an authenticated channel.

The security notion for these authentication protocols is adopted from a model proposed by Bellare-Rogaway [2], [3]. Let us say that the devices A and B aim to authenticate some information K_A and K_B, to each other, respectively. K_A and K_B can be the permanent public keys or ephemeral Diffie-Hellman components of A and B. We will consider an adversary $\mathcal{A}$ against the authentication protocol, which is allowed to launch the protocol between A and B on inputs (K_A and K_B) of his choice. In other words, using the launch queries, $\mathcal{A}$ can trigger the protocol between A and B on input K_A and K_B, respectively. The challenger (against which the adversary plays the game) responds by initializing the state of the invoked session and sending back to $\mathcal{A}$ the message it generates. The adversary can also issue send queries for any previously initialized session on a message M as input, which triggers the challenger to deliver message M to that particular session and respond by following the protocol on its behalf.

Eventually, the adversary is said to succceed in the game or in breaking the protocol if (1) A accepts K'_B as authentic although a session with K'_B on B was never launched by the adversary, or (2) B accepts K'_A as authentic although a session with K'_A on A was never launched by the adversary, or (3) both (1) and (2) satisfy.

Theorem 1: If $h()$ is a strongly-collision resistant hash function, then the VIC protocol securely authenticates K_A to B, and securely authenticates K_B to A.

Proof: Intuitively, the security of VIC depends on the

Y has $\rightarrow$ X has $\downarrow$	Camera and display	Camera only	Display only	None
Camera and Display	$X \leftrightarrow$ Y	$X \leftrightarrow$ Y$_s$	$X \leftarrow$ Y $X \xrightarrow{p} Y$	$X \leftarrow Y_s$
Camera only	$X_s \leftrightarrow Y$	$X_s \leftrightarrow Y_s$	$X \leftarrow Y$ $X \xrightarrow{p} Y$	$X \leftarrow Y_s$
Display only	$X \rightarrow Y$ $X \xleftarrow{p} Y$	$X \rightarrow Y$ $X \xleftarrow{p} Y$	none	none
None	$X_s \rightarrow Y$	$X_s \rightarrow Y$	none	none

Notation:
P_s: "Device P needs a static barcode label affixed to it."
$P \rightarrow Q$: "Device P can strongly authenticate to device Q."
$P \xrightarrow{p} Q$: "Device P can demonstrate its presence to device Q."

Fig. 2. Types of authentication achievable using SiB for given device type combinations

1) $A \longrightarrow B$ (insecure channel): K_A
2) $A \longleftarrow B$ (insecure channel): K_B
   A calculates h_A as $h(K_A, K_B)$ and B calculates h_B as $h(K_A, K_B)$
3) $A \longrightarrow B$ (visual channel): h_A
   B compares h_A and h_B. If they match, B accepts and continues. Otherwise B rejects and aborts. In either case, B indicates accept/reject to the user.
4) A prompts user as to whether B accepted or rejected. A continues if the user answers affirmatively. Otherwise A rejects.

Fig. 3. VIC mutual authentication protocol

Y has $\rightarrow$ X has $\downarrow$	Camera and display	Camera only	Display only	None
Camera and Display	X $\leftrightarrow$ Y	X $\leftrightarrow$ Y	X $\leftrightarrow$ Y	none
Camera only	X $\leftrightarrow$ Y	none	X $\leftrightarrow$ Y	none
Display only	X $\leftrightarrow$ Y	X $\leftrightarrow$ Y	none	none
None	none	none	none	none

Notation:
P $\leftrightarrow$ Q: "Devices P and Q can strongly authenticate each other."

Fig. 4. Types of authentication achievable using VIC for given device type combinations.

attacker not being able to find two numbers K'_A and K'_B such that $h(K_A, K'_B) = h(K'_A, K_B)$ (this is the acceptance condition). This is because if the attacker can find such values, then he can modify K_A to K'_A, and K_B to K'_B, during the protocol, and succceed in authenticating K'_A and K'_B to B and A, respectively. A formal proof of this argument, based on the contrapositive argument, is discussed as follows.

Basically, we show that if there exists an adversary $\mathcal{A}$ who can break the VIC protocol, then we can construct an adversary $\mathcal{B}$ (a simulator) who can break the collision-resistance of the underlying hash function $h()$. To start with, A launches an instance of the protocol between A and B with inputs K_A and K_B, respectively. When A transmits the value K_A to B, $\mathcal{A}$ simply drops it and inserts a different value K'_A instead. Similarly, when B transmits the value K_B to A, $\mathcal{A}$ drops it and inserts a different value K'_B instead. $\mathcal{A}$ delievers the value $h_A = h(K_A, K'_B)$ as it is to B over the OOB channel (note that A can not modify values transmitted over the OOB channel). Since $\mathcal{A}$ succeeds in breaking the protocol, K'_A is accepted by B as authentic, i.e., $h(K'_A, K_B) = h_A$. This means that the values (K'_A, K_B) and (K_A, K'_B) collide with each other, which the adversary $\mathcal{B}$ produces as an output, thus breaking the collision property of the $h()$. ■

IV. Seeing With Less: Visual Channel in Constrained Devices

Now we turn our attention to the third drawback of SiB. In this section, we show how to enable visual channel authentication on devices with very limited (or tiny) displays and in the minimal case, with extremely constrained displays

consisting of only single light source (or LED). These extensions are made possible by using key agreement protocols that require short authenticated integrity checksums. We begin by describing such protocols.

A. Authentication Using Short Integrity Checksums

The reason why SiB needs good displays is the high visual channel bandwidth required for the SiB protocol. Assuming that the attackers have access to today's state-of-the-art computing resources, the bandwidth needed is at least 48 bits in the case of ephemeral keys [8], rising to 80 bits in the case of long-lived keys. These numbers can only increase over time.

Fortunately, there is a family of authentication protocols that has very low bandwidth requirements. The first protocols in this family, proposed by Gehrmann et al. in [7], [8], were aimed at using the human user as the authentication channel; hence the name "Manual authentication (MANA)". The MANA protocols are based on OOB channels which are assumed to be both authenticated and secret. Several subsequent protocols, based only on authenticated OOB channels, have been reported [12], [29], [13], [17]. We apply the variation called "MA-3" [13] to get VICsh (VIC with short checksum) as shown in Figure 5[4]:

K_A, K_B are as in the case of SiB. $h()$ represents a commitment scheme and $hs()$ is a mixing function with a short n-bit output (e.g., n = 15 ... 20) such that a change in any input bit will, with high probability, result in a change in the output. In practice, $hs()$ can be the output of a cryptographic hash function truncated to n bits. Refer to [13] for formal description of the requirements on $h()$ and $hs()$, and their instantiations, as well as for the proofs of security of the protocol. Informally, the security of the protocol depends on the following:

- neither party reveals the value of its random bit string (R_A or R_B respectively) until the other party commits to its own random bit string, and
- each party knows that the public data (K_A and K_B) used in the computation of the check-value (hs_A or hs_B) is known to it before it reveals its random bit string.

Suppose the man-in-the-middle attacker has a public key K_M. To fool device A into accepting K_M as B's public key, the attacker needs to ensure that $hs_A = hs(R_A, X, K_A, K_M)$ and $hs_B = hs(Y, R_B, Z, K_B)$ are equal. The attacker can choose K_M, X, Y and Z, but he must make his choices before knowing R_A or R_B. Therefore, whatever his strategy for choosing the values, the chance of success is $x = 2^{-n}$. Similarly, the probability of the attacker fooling device B into accepting K_M as A's public key is also x. More importantly, this probability does not depend on the computational capabilities of the attacker, as long as $h()$ is secure.

To summarize, below are the following main differences between SiB and VICsh:

1) SiB requires transmission of at least 68 bits of data over the visual OOB channel, whereas VICsh only requires 15-20 bits of data over the OOB channel.
2) SiB requires both devices to be equipped with cameras. If one of the devices does not possess the camera, it is not possible to achieve mutual authentication. VICsh, on the other hand, can be used to provide mutual authentication even when one of the devices lack a camera.

[4]We chose MA-3 over the protocol in [29] for reasons of efficiency because MA-3 requires fewer rounds of communication over the insecure channel.

B. Trimming Down the Display

Armed with the variation of VIC described above, we are now ready to investigate visual channel authentication on devices with very limited displays. Recall that our motivation is to support visual channel authentication on various commercial devices, such as wireless access points, Bluetooth headsets, etc. These devices typically have only the most limited form of a display consisting of a single bi-state light source, such as a single light-emitting diode (LED). In this section, we describe each aspect of the realization of single LED based visual channel authentication.

Transmission: We use frequency modulation to encode the data being transmitted (see Figure 6). The sender turns the light-source on and off repeatedly. The data is encoded in the time interval between each successive "on" or "off" event: a long gap represents a '1' and a short gap represents a '0'. Since the channel is unidirectional, the transmitter cannot know when the receiver starts reception. Therefore, the transmitter keeps repeating the sequence until either the user approves the key agreement, or a timeout occurs.

The camera phones of today are limited to a frame rate of about 10 video frames/second, and as we are receiving the bits with frequency modulation without synchronization, we are bound by the Nyquist-Shannon sampling theorem (sampling rate = 2 $\times$ bandwidth for no loss of information) [16]. This limits the transfer speed with this algorithm to around 5 bits/second.

Reception: The receiver processing is analogous: simplified, each received video frame is compressed into one value per frame (the sum of all the pixel values)[5], and the first-order difference between consecutive values (i.e., the derivative) is compared against a relative threshold based on maximum observed variation in the pixel sum. If the derivative is steep enough and in the right direction (alternating between positive and negative) a transition in lighting is registered. The time between two consecutive changes indicates the transfer of either a '1' or a '0' bit as depicted in Figure 6.

Dealing with Errors, and Trading Efficiency with Security: We designed two mechanisms that allow the possibility of a parameterizable trade-off between execution time and the level of security.

First, the data being transmitted via the visual channel, i.e., the integrity checksum, is known to the receiver in advance. We use this simple observation to reduce execution time.

[5]The fact that the video frame is collapsed into one value per frame also shows the feasibility of using a sensitive light sensor combined with an analog-to-digital converter as a cheaper form of receiving device – with no change to the algorithms described in the paper. We have left the implementation of such a receiver as future work.

1) A chooses a long random bit string R_A and calculates h_A as $h(R_A)$.
   $A \longrightarrow B$ (insecure channel): h_A, K_A
2) B chooses its own long random bit string R_B
   $A \longleftarrow B$ (insecure channel): R_B, K_B
3) $A \longrightarrow B$ (insecure channel): R_A
   B now computes h'_A as $h(R_A)$ and compares it with the h_A received in message 1. If they do not match, B aborts. Otherwise B continues.
4) A calculates hs_A as $hs(R_A, R_B, K_A, K_B)$ and B calculates hs_B as $hs(R_A, R_B, K_A, K_B)$
   $A \longrightarrow B$ (visual channel): hs_A
   B compares hs_A and hs_B. If they match, B accepts and continues. Otherwise B rejects and aborts. In either case, B indicates accept/reject to the user.
5) A prompts user as to whether B accepted or rejected. A continues if the user answers affirmatively. Otherwise A rejects.

Fig. 5. VICsh mutual authentication protocol based on short integrity checksum

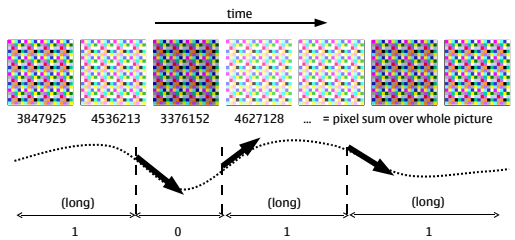


Fig. 6. Data transmission via a single light-source visual channel

Recall that the sender repeats the n-bit string a number of times. The receiver proceeds in the following way: reception may start at any bit position, and the receiver records until the n-bit tail of the received bit-string matches against any of the rotated versions of the expected n-bit string. Therefore, the receiver accepts at most n possible matches for the transmitted value. For example, if the transmitted string is '1011', the receiver accepts if it receives any of the strings '1011', '0111', '1110', '1101'.

Second, rather than doing error correction, we tolerate (or simply accept) a certain number of errors in the n-bit transmission. With k accepted errors, the number of possible matches, based on a binomial distribution of errors, is $\sum_{i=0\ldots k} \binom{n}{i}$.

Using these mechanisms the probability that the receiver will accept a random string as valid will increase from the original value of $p = \frac{1}{2^n}$. Accounting for both modifications we can estimate an upper bound to

$$p = n \; \frac{\sum_{i=0}^{k} \binom{n}{i}}{2^n}$$

The given bound allows us to get an idea of the degree of loss of security. If e.g. $k = 3$ bits are allowed to be wrong in an $n = 24$ bit sequence, p is 0.0064, whereas if only 1 bit error is allowed, p is 0.00004.

For personal use, e.g., when a user wants to pair his workstation with his own wireless access point, an attack success probability of 0.00004 is acceptable. In other situations where, say, every day thousands of pairings are done with a device located in a public space, the attack success probability needs to be lower.

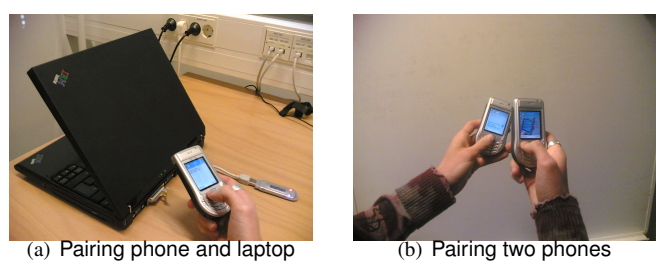
There are several ways to trade off security and execution time. The attack success probability p can be decreased by:

- increasing the length of the checksum n,
- reducing the number of acceptable errors k,
- reducing the number of possible rotations that are acceptable as matches (say only every fourth)
- adding an external end marker to the protocol (e.g., the light-source staying "on" for 0.5 seconds) to indicate when it starts to repeat the checksum string, bringing the attack success probability down to $\frac{\sum_{i=0}^{k} \binom{n}{i}}{2^n}$.

Applying one or several of these measures will result in changed lower and upper "bounds" for the execution time.

V. Implementation

We have developed two proof-of-concept prototypes for our method: one where a single blinking LED (connected to the parallel port of a PC) sends a signal that is received by a camera phone and the other where the flashing screen of one phone is recorded by another phone. Figures 7(a) and 7(b) illustrate our two demonstrator implementations. In 7(a), a Bluetooth pairing is established between a Symbian 8.0 camera phone (Nokia 6630) and a Linux laptop with an LED (illustrating, e.g., a wireless access point). In 7(b), two (Nokia 6630) phones are paired using the display of one phone as the bi-state light.



(a) Pairing phone and laptop (b) Pairing two phones

Fig. 7. Scenarios for the proof-of-concept implementation

Our algorithm makes bit reception quite tolerant. The data can be received at a distance of several tens of centimeters, the implementation is agnostic to camera focus problems and tolerates a fair bit of camera shaking, turning, etc. The real-time progress of the matching is indicated at runtime on the handset screen by displaying two parameters: percentage of the string successfully received so far and a related confidence

level. In our setup, we used a 24-bit checksum with 1 error accepted.

Figure 8 gives a more detailed description of the user interface of our Symbian implementation during pairing with the laptop. In Figure 8(a), the user starts the pairing from a menu.[6] In Figure 8(b), the phone scans the Bluetooth neighborhood and finds the laptop. In Figures 8(c) and 8(d), the phone starts recording with its camera and the user positions the phone so that the blinking of the LED is shown in the viewfinder. The recording status is updated in the viewfinder in real-time. In 8(e), the pairing is complete for the phone once the correct checksum has been received and accepted. The success is reported to the user, who is instructed to accept the pairing at the access point to achieve mutual authentication.

VI. Usability Testing

In order to evaluate the proposed method, we pursued a usability study. We tested our method for the use case of pairing two (Nokia 6630) cell phones, as depicted in Figure 7(b). The goal of our usability study was to test our method with respect to the following factors:

1) Efficiency: how long the method takes (i.e., time-to-completion).
2) Usability: how the method fares in terms of user burden (i.e., ease-of-use perception).

Study Participants: We recruited 20 subjects[7] for our study, which lasted over a period of more than two weeks. Subjects were chosen on a first-come first-serve basis from respondents to recruiting posters and email ads. Prior to recruitment, each subject was briefed on the estimated amount of time required to complete the test. We prepared two questionnaires: *background* – to obtain user demographics and *post-test* – for user feedback on method tested.

Recruited subjects were mostly university students, both graduate and undergraduate. This resulted in a fairly young (ages between 18-29), well-educated and technology-savvy[8] participant group.

None of the study participants reported any physical impairments that could have interfered with their ability to complete given tasks. The gender split was: 65% male and 35% female. Gender and other information was collected through *background questionnaires* that all subjects completed prior to testing.

Testing Process: Our study was conducted in a variety of campus venues of our University including, but not limited to student laboratories, cafés, student dorms/apartments, classrooms, office spaces and outdoor terraces. This was possible since the test devices were mobile, test set-up was more-or-less automated and only a minimal involvement from the test administrator was required.

[6]The pairing must be initiated also from the laptop side. The rationale for this is explained in Section VII-B.

[7]It is well-known that a usability study performed by 20 participants captures over 98% of usability related problems [6].

[8]All participants were regular computer users with at least one wireless personal device.

After giving a brief overview of our study goals we asked the participants to fill out the background questionnaire in order to collect demographic information. In this questionnaire, we also asked the participants whether they suffer(ed) from any visual impairment, or have any condition that may interfere with their holding objects steady or their reflexes. Next, the participants were given a brief introduction to the cell-phone devices used in the tests.

We created one test case where the receiving device always receives the video captured with the help of the user and always accepts it as legitimate. The same test case was repeated twice per user. This simulated a real testing scenario for our method. By repeating a test case twice, we wanted to figure out whether "learning" would have an impact on the performance of the method.

Each participating user was next given the two devices (Nokia 6630) and asked to follow on-screen instructions shown before each task to complete it. User interactions throughout the test and timings were logged manually by the test administrator. After completing the tasks each user filled out a post-test questionnaire form, where they provided their feedback on the method tested. The users were also given a few minutes of free discussion time, where they explained to the test administrator about their experience with the method they tested.

Test Results and Interpretations: We collected data in two ways: (1) by timing and logging user interaction, and (2) via questionnaires and free discussions.

For the tested method, completion times, errors and actions were managed by the test administrator. We recorded the timing from the start of the method to its very end. The timing information is graphed in Figure 9. The average time taken by our users was 46.95 seconds (standard deviation 4.39 seconds) in the first run and 38.85 seconds (standard deviation 1.87 seconds) in the second run. As the results show, although the video transmission time was the same in two test executions, each user took shorter to complete the whole process in the second execution. Our findings were confirmed using the paired t-tests, which indicated that there is a statistically significant difference ($p < 0.0001$) in the timings of the two executions (the second execution being significantly faster). This clearly indicates that the test timings will improve as users become more and more familiar with the method. In all test runs, users correctly accepted the result of pairing on the blinking device as indicated by the capturing device.

We note that our pairing speed of around 40 seconds compares favorably to the speed of another pairing method [24], which is a leading alternative suitable for devices with constrained interfaces (such as access points, headsets). We will further compare the two methods, in terms of their efficiency, security and usability, in Section VIII. We note, however, that pairing transmission time can be further reduced by making use of multiple LEDs, which are often available on commodity devices (e.g., access points) or can be added at a little additional cost.

Based on the unpaired t-tests, we did not find any statistically significant effect of age and gender on the executions timings.

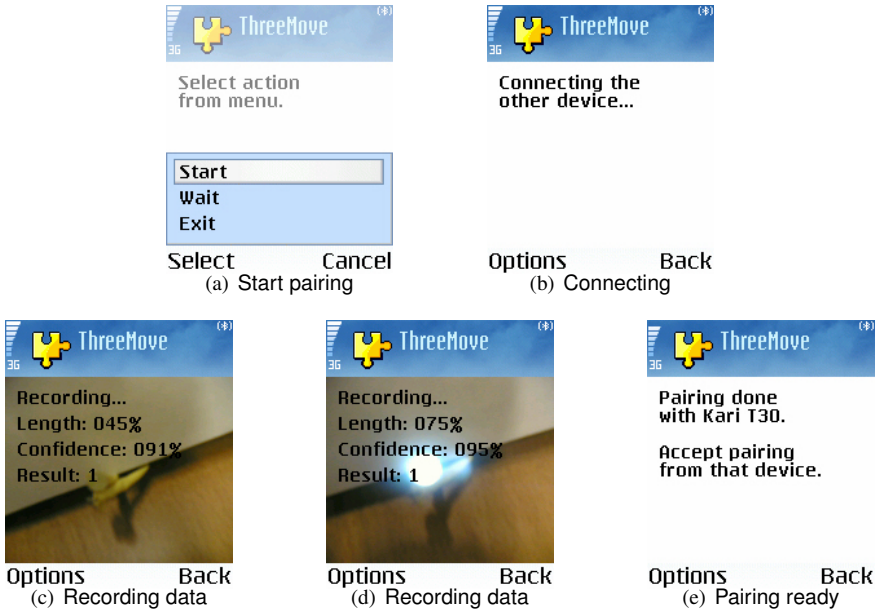


Fig. 8. Screen-shots from the Symbian implementation

In the post-test questionnaire, we solicited user opinions about the tested method. Participants rated the method for its ease-of-use: very easy, easy, hard or very hard. The ease-of-use ratings are graphed in Figure 10. As our results show, most users found the method fairly easy to operate. Moreover, when asked whether the method was professional, 80% of our participants responded affirmatively.

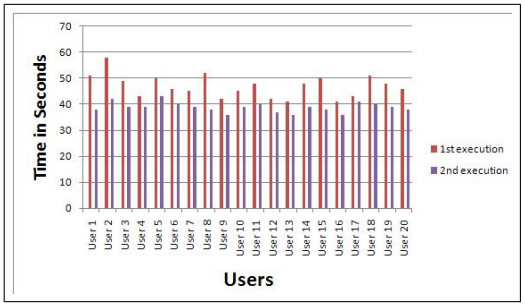


Fig. 9. Time-to-Completion for Successful Pairing

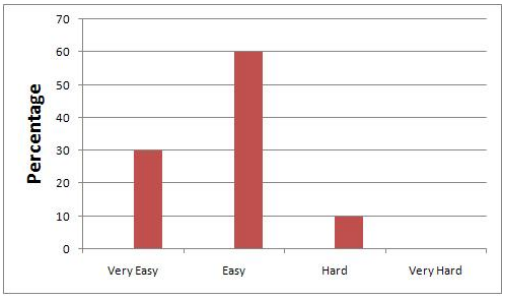


Fig. 10. Ease-of-Use Ratings

VII. Discussion

In this section we discuss the applicability of our results, examine practical use cases, and discuss practical issues like device discovery.

A. Comparison of Different Protocols

Table 11 summarizes our recommendations on how mutual authentication can be achieved with different device type combinations. If both devices have camera and display, mutual authentication can be achieved either using SiB or VIC. SiB can be used with camera-only devices which can have static barcodes affixed to them. The case of two display-only devices is out of scope for this paper, and the basic MANA techniques which require the user to visually compare two short strings [7], [8] can be used. In all the other cases, VIC could be the best choice since it provides mutual authentication and good usability, as our results indicate.

Y has $\rightarrow$ X has $\downarrow$	Camera and display	Camera only	Display only
Camera and Display	SiB/VIC	VIC	VIC
Camera only	VIC	SiB[a]	VIC
Display only	VIC	VIC	MANA

[a]Both devices need static barcode labels affixed to them.

Fig. 11. Recommended protocol to achieve mutual authentication for given device type combinations

Since the bandwidth requirement for VICsh protocol is low, this protocol could be used in scenarios where it is not possible to reach the bandwidth required by the VIC protocol. One example of such a scenario is a WLAN access point that is mounted high up on the wall or ceiling. It is not possible to read the barcode affixed to such an access point with the current camera phones, but it might be possible to read the "blinking" of the access point if the light source is powerful enough.

B. Device Discovery Strategies

Previous proposals on security initialization using out-of-band methods [26], [1] have argued that one of the main benefits of using an out-of-band channel for security initialization is the fact that device discovery is part of the OOB message exchange. For example in the approach proposed by Balfanz et al. [1] the devices exchange complete addresses over infrared, and thus no in-band device discovery is needed. In SiB approach, the device discovery is done manually (because current phones can not display big enough bar codes to contain both the address and the hash of a public key), but the authors state that the optimal solution would be to encode both the address and the public key hash to the bar code.

We argue that in many scenarios an in-band device discovery is actually needed before the OOB message exchange. The increasing number of different OOB channels (such as infrared, camera and full display, camera and single LED etc.) results in situations where the user might not always know which OOB to use with the two particular devices at hand. For example a user wanting to pair a camera phone (camera, display, no infrared) with a laptop (infared, display, no camera) might be confused about the different OOB possibilities. It should not be the user's burden to figure out which OOB to use (and how), but instead an in-band device discovery should take place and the best mutually supported OOB channel should be negotiated in-band and the user should be guided to use this OOB. Negotiations must be protected against bidding-down attacks in the usual manner, by having the parties exchange authenticated confirmations of the negotiation messages once key establishment is completed (as is done with the "Finished" message in TLS[5]). As long as the chosen authentication mechanism can not be broken in real-time, attempts to bid-down will be detected by this check.

In order to conveniently discover the desired device in-band, the user must put one of the devices into a temporary special discoverable mode so that the user does not have to select the correct device from a long list of (probably meaningless) device names. We call this action *user conditioning*. From the user's point of view this action can be performed, e.g., by pressing a button on the device or by selecting a menu option.

Not all bearers support in-band discovery without manual device selection. Likewise, pure out-of-band discovery is not always feasible with constrained OOB channels. In these cases, the constrained OOB can be used to improve the usability of the in-band discovery process. A device can, e.g., send the last 10 bits of its address over OOB. At the same time the other device can scan and automatically discard devices whose address does not match these 10 bits. With high probability the correct device can be selected automatically and the user does not have to be presented a list of device names.

C. Denial-of-Service

One concern in device pairing is the possibility of a denial-of-service attack. An attacker can disrupt a pairing attempt between two devices by simultaneously invoking pairing with one or both of the same devices. More concretely, during the device discovery phase, one of the pairing devices transmits – over the wireless channel – its device identifier to the other device; the adversary could also insert its own device identifier and likely fool the receiving device into initiating pairing with its (the adversarial) device. Accidental simultaneous pairing is likely to be very rare because of the user conditioning described in Section VII-B. Thus, if a device detects multiple pairing attempts, the best strategy may be to ask the user to try again later, rather than ask the user to choose the correct device. Another possibility to detect multiple parallel device pairing attempts is by sending (part or whole of) the device identifier over the visual channel (i.e., via blinking LED), as discussed in Section VII-B. This way the receiving device will only establish pairing with the device having an identifier, or a part thereof, that matches with the value received over the visual channel. This in turn helps prevent the above denial-of-service attack with a high probability. We note, however, that in wireless networks, elaborate attempts to protect the pairing protocol against malicious attempts of denial-of-service are not cost effective because an attacker can always mount denial-of-service by simply disrupting the radio channel.

VIII. Comparison with Related Work

A great deal of work has been done in the area of device pairing. One of the earliest techniques – "Resurrecting Duckling", proposed by Stajano and Anderson – was to have two devices share a secret using a physical connection such as a cable [27]. Unfortunately, this solution does not apply to many pairing scenarios, such as when the two devices lack a common interface or a proper cable is unavailable. Balfanz et al. took this idea one step further by replacing the physical connection with an infrared channel [1]. Their pairing protocol, called "Talking to Strangers", required the pairing devices to exchange public keys over a wireless channel, then swap 80 bit (or larger) hashes of their respective keys through the infrared (IR) channel. This setup's central shortcoming is that it is only designed for devices with infrared transceivers. Note that the VIC protocol that we presented in Section III can be applied to this method; this will mean that one device needs to have an IR transmitter and the other, an IR receiver.

A different approach to the problem of pairing is to use a wireless channel to perform a key exchange, then form visual channels by which the device user can manually compare the shared secret on each device being paired. Since this comparison process creates a high burden for the users of the devices, several techniques were created to visualize the key data in a manner more conducive to comparison. A few notable examples based on Image Comparison include Goldberg's Snowflake [9] and Random Arts visual hash of Perrig and Song [18]. These methods involve complex visualizations and as such are only applicable to devices with high-resolution displays such as laptops.

Goodrich et al. proposed "Loud & Clear" by extending the idea of Balfanz et al. to create a pairing system based on the idea of "Mad Libs" word puzzles [10]. This pairing setup works through channels formed by random English Phrases. The pairing devices exchange public keys, then independently hash both of them. Each device then encodes these hashes into a random English sentence. Both devices convey these

sentences to the device user either in verbalized form through a speaker or written form through a display. The user compares these sentences to determine if the exchanged keys differ. As proposed, this system calls for four channels and two manual comparisons by the user of the devices. This scheme can be modified, as we proposed in Section III, in a simple way to improve the pairing experience for the user. Instead of hashing the two keys and encoding them as separate sentences, the devices can concatenate both keys, then hash and encode the resulting value (of at least 160 bits). This scheme would thus require a single comparison as opposed to two. While this scheme does not rely on any specific receiver, it does require both devices to have a display or a speaker.

A usability study in which various simple pairing schemes were compared was carried out by Uzun et al. [28]. Their tests assume devices with displays capable of displaying 4 decimal digits as transmitters. Three types of pairing approaches are analyzed. The first, "Compare-and-Confirm," only requires users to read and compare the data displayed on each device. The second is called "Select-and-Confirm" and asks that users select a particular 4-digit value from a set of such values stored on one device that matches a single value displayed on the other device. The final technique, "Copy-and-Confirm" asks users to read the value shown on the first device and input it on the second device. Note that Copy-and-Confirm and Select-and-Confirm are based on our unidirectional OOB protocol – VICsh – that we presented in Section IV.

Some recent work has focused upon pairing devices which possess constrained interfaces. These include the BEDA scheme [24], which requires the users to transfer the OOB checksums from one device to the other using "button presses;" the schemes [19], [21], which require the users to compare simple blinking or beeping patterns on two devices.

The BEDA proposal achieves device pairing through manual "button presses" [24]. The underlying idea of this protocol is to carry out a Diffie-Hellman key exchange then authenticate the agreed-upon key using a brief password. This password is established using one of three options, each of which in based on button presses. Variant one ("Button-Button") asks the device user to press a button simultaneously on each device. This press must occur within a certain time interval, and each press is manipulated to produce 3 bits of password data. A 15-bit password can therefore be established using 5 button presses. Variants two and three ("LED-Button" and "Vibrate-Button") involve one of the devices being paired computing a short password. This password is encoded in 3-bit blocks as a delay between OOB outputs from the device. When the transmitting device flashes or vibrates, the device's user presses a button on the other device to transfer over the password.

A problem with the above system is that secrecy of the negotiated password is essential to the scheme's security. If the button presses or OOB output used during the pairing procedure are observed or recorded then the secrecy of the password, and thus the system's security, cannot be guaranteed. On the positive side, though, this setup can easily integrate one of the short OOB protocols [13], [17] and is also useful for pairing devices which lack quality transmitters or receivers. With the minimum possible interfaces, namely a LED and a button on each, the devices can take turns exchanging their checksums by having one device blink its LED and the other accept correspondingly timed button presses. This would unfortunately take a great deal of time. Transmitting a 15-bit checksum value would take a minute in either direction (as per the results presented in [24]) because it takes users at least 3 seconds to press a button in response to a visual stimulus. The unidirectional pairing protocol, VICsh, discussed in this paper (in Section IV) could be applied to eliminate the need to transfer OOB data in both directions and therefore bring the execution time close to a minute. Note that even this variant would be slower compared to the camera-based pairing method that we presented in this paper (the latter takes around 40 seconds of pairing time as indicated by our results in Section VI). Moreover, our method, being automated, would be less cumbersome for the users.

In [19], authors developed a pairing method based on synchronized audio-visual patterns. Proposed methods, "Beep-Beep", "Blink-Blink" and "Beep-Blink", involve users comparing very simple audiovisual patterns, e.g., in the form of "beeping" and "blinking", transmitted as simultaneous streams, forming two synchronized channels. One advantage of these methods is that they require devices to only have two LEDs or a basic speaker. However, as discussed in [19], these methods are susceptible to human errors since they are based on careful manual comparison. This is in contrast to the camera-based pairing method we proposed in this paper, which is less likely to result in human errors. Most recently, the approach of [19] was extended by making use of an auxiliary device, such as a smartphone [23]. This reduces the likelihood of manual errors, however, at the cost and complexity of introducing a third device to pair two devices.

Working independently, Roth et al. [21]developed a scheme similar to the Blink-Blink scheme of [19]. This system is designed to protect against "evil twin" public access points. The two protocols, [21] and [19], differ significantly both in terms of implementation and user experience, however. In the protocol of Roth et al. the user is given control over the interval in which each bit of OOB output is compared. This control is performed by pressing and releasing a button on the pairing devices. This can be contrasted with the Blink-Blink system of [19], which instead uses a static, experimentally predetermined interval for the comparison of SAS bits. Similarly, the evil twin detection scheme specifies that the transmission of each data bit be triggered by a signal sent by the other pairing device over a wireless channel. Thus the device user must verify whether the k wireless signals sent to facilitate the transmission of a k-bit SAS value were delayed or attacked in some other manner. This differs significantly from the Blink-Blink setup of [19], which only relies on one wireless channel synchronization signal. Similar to [19], the method of [21] is susceptible to human errors since it is based on careful manual comparison.

Some follow-on work (HAPADEP [25]) considered pairing devices that – at least at pairing time – have no common wireless channel. HAPADEP uses pure audio to transmit cryptographic protocol messages and requires the user to merely

monitor device interaction for any extraneous interference. It requires both devices to have speakers and microphones. To appeal to more realistic settings, this proposal can be based on the VICsh protocol we presented in this paper (depicted in Section IV). This HAPADEP variant would use the wireless channel for cryptographic protocol messages and the audio as the OOB channel. In it, only one device would need a speaker and the other –a microphone. Also, the user would not be involved in any comparisons.

IX. Conclusions

In this paper, we proposed several improvements and extensions to the recently proposed approach of using a visual channel to implement secure pairing. We showed how strong mutual authentication can be achieved using just a unidirectional OOB channel, which could also improve the usability of the pairing process.

We then showed how visual channel authentication can be used even on devices that have very limited displaying capabilities, such as a single LED. Commoditized devices like wireless access points, and devices with form factor limitations like headsets, cannot afford to have full displays capable of displaying barcodes. Our contribution makes it possible to use visual channel authentication even on such devices. We also evaluated our method via a usability study. The results of our study indicate that the method is suitable for ordinary users with reasonable execution times and also that these timings can be sped up as users become more and more familiar with the method.

It would be feasible to trim down the camera to a simple light sensor. Although at first glance this might seem to be the same as a one-way infrared communication channel, there are important differences in terms of user perception and cost first, a user can easily see a light source, and can detect the presence of a false source; second, adding an infrared interface for the purpose of secure device pairing is not an economically viable option for commodity devices like wireless access points or Bluetooth headsets; but typically they tend to have one or more LEDs which can be used to implement the technique we propose. By integrating a flashing light-source on one device and a light sensor on another, two wireless sensor devices can thus be efficiently paired.

Acknowledgements

We are grateful to Adrian Perrig, who shepherded the conference version of this paper, Jonathan McCune, Markku Kylänpää, and the anonymous reviewers for their thoughtful and constructive feedback which helped us improve the paper. We also thank Marie Selenius, Dr. Niklas Ahlgren, and Dr. Valtteri Niemi for insights into the counting argument, and Kaisa Nyberg and Stanisław Jarecki for valuable feedback on our protocols. Finally, we thank Arun Kumar for administering our usability study.

References

[1] Dirk Balfanz, Diana Smetters, Paul Stewart, and H. Chi Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Network and Distributed System Security Symposium*, 2002.

[2] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *CRYPTO '93: Proceedings of the 13th annual international cryptology conference on Advances in cryptology*, pages 232–249, 1994.

[3] Mihir Bellare and Phillip Rogaway. Provably secure session key distribution: the three party case. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 57–66, 1995.

[4] RVSI Acuity CiMatrix. Data Matric Barcodes, 2005. Available at http://www.rvsi.net/.

[5] Tim Dierks and Christopher Allen. The TLS protocol version 1.9. Internet Engineering Task Force, RFC 2246, January 1999.

[6] L. Faulkner. Beyond the five-user assumption: Benefits of increased sample sizes in usability testing. *Behavior Research Methods, Instruments, & Computers*, 35(3):379–383, 2003.

[7] Christian Gehrmann et al. SHAMAN Deliverable: Detailed Technical Specification of Mobile Terminal System Security, May 2002. Available at www.isrc.rhul.ac.uk/shaman/docs/d10v1.pdf.

[8] Christian Gehrmann, Chris J. Mitchell, and Kaisa Nyberg. Manual authentication for wireless devices. *RSA CryptoBytes*, 7(1):29 – 37, Spring 2004.

[9] Ian Goldberg. Visual Key Fingerprint Code, 1996. `http://www.cs.berkeley.edu/iang/visprint.c`.

[10] Michael T. Goodrich, Michael Sirivianos, John Solis, Gene Tsudik, and Ersin Uzun. Loud and Clear: Human-Verifiable Authentication Based on Audio. In *International Conference on Distributed Computing Systems (ICDCS)*, 2006.

[11] Stephen R. Hanna. Configuring Security Parameters in Small Devices, July 2002. draft-hanna-zeroconf-seccfg-00.

[12] Jaap-Henk Hoepman. The ephemeral pairing problem. In *Financial Cryptography*, number 3110, 2004.

[13] Sven Laur, N. Asokan, and Kaisa Nyberg. Efficient mutual data authentication based on short authenticated strings. IACR Cryptology ePrint Archive: Report 2005/424 available at `http://eprint.iacr.org/2005/424`, November 2005.

[14] Anil Madhavapeddy, David Scott, Richard Sharp, and Eben Upton. Using camera-phones to enhance human-computer interaction. In *Sixth International Conference on Ubiquitous Computing*, 2004.

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

[16] Harry Nyquist. Certain topics in telegraph transmission theory. *Transasctions of the American Institute of Electrical Engineers (AIEE)*, 47:617–644, 1928.

[17] Sylvain Pasini and Serge Vaudenay. An optimal non-interactive message authentication protocol. In *CT-RSA*, 2006 (to appear).

[18] Adrian Perrig and Dawn Song. Hash visualization: a new technique to improve real-world security. In *Cryptographic Techniques and E-Commerce (CrypTEC)*, 1999.

[19] Ramnath Prasad and Nitesh Saxena. Efficient device pairing using human-comparable synchronized audiovisual patterns. In *Applied Cryptography and Network Security (ACNS)*, 2008.

[20] Michael Rohs and Beat Gfeller. Using camera-equipped mobile phones for interacting with real-world objects. In Alois Ferscha, Horst Hoertner, and Gabriele Kotsis, editors, *Advances in Pervasive Computing*, 2004.

[21] Volker Roth, Wolfgang Polak, Eleanor Rieffel, and Thea Turner. Simple and effective defenses against evil twin access points. In *ACM Conference on Wireless Network Security (WiSec), short paper*, 2008.

[22] Nitesh Saxena, Jan-Erik Ekberg, Kari Kostiainen, and N. Asokan. Secure device pairing based on a visual channel. In *IEEE Symposium on Security and Privacy, appeared as a short paper (6 pages)*, 2006.

[23] Nitesh Saxena, Jonathan Voris, and Borhan Uddin. Universal Device Pairing Using an Auxiliary Device. In *Symposium On Usable Privacy and Security (SOUPS)*, July 2008.

[24] Claudio Soriente, Gene Tsudik, and Ersin Uzun. BEDA: Button-Enabled Device Association. In *International Workshop on Security for Spontaneous Interaction (IWSSI)*, 2007.

[25] Claudio Soriente, Gene Tsudik, and Ersin Uzun. Hapadep: Human asisted pure audio device pairing. ISC, 2008.

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

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

[28] Ersin Uzun, Kristiina Karvonen, and N. Asokan. Usability analysis of secure pairing methods. In *Usable Security (USEC)*, 2007.

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

[30] Simon Woodside. Read real-world hyperlinks with a camera phone, February 2005. Available at http://semacode.org.