Loud and Clear: Human-Verifiable Authentication Based on Audio

Michael T. Goodrich, Michael Sirivianos, John Solis, Gene Tsudik, and Ersin Uzun
University of California, Irvine
{goodrich,msirivia,solis,gts,euzun}(at)ics.uci.edu

Abstract

Authentication of communication channels between devices that lack any previous association is an challenging problem. It has been considered in many contexts and in various flavors, most recently, by McCune et al., where human-assisted device authentication is achieved through the use of photo cameras (present in some cellphones) and 2-dimensional barcodes. Their proposed Seeing-is-Believing *system allows users with devices equipped with cameras to use the visual channel for authentication of* unfamiliar *devices, so as to defeat man-in-the-middle attacks.*

In this paper, we investigate an alternative and complementary approach—the use of the audio channel for human-assisted authentication of previously un-associated devices. Our motivation is three-fold: (1) many personal devices are not equipped with cameras or scanners, (2) some human users are visually impaired (hence, cannot be in the authentication pipeline of a vision-based system), and (3) some usage scenarios preclude either taking a sufficiently clear picture and/or the use of barcodes.

We develop and evaluate a system we call **Loud-and-Clear (L&C)** *authentication, which, like Seeing-is-Believing, places little demand on the human user. The L&C system is based on the use of a text-to-speech engine to read an auditorially-robust, grammatically-correct pass-phrase derived from an authentication string that is to be used by peer devices. In particular, by coupling the auditory reading of the one-way hash of an authentication string on one device with the display of of this text on another device, we demonstrate that L&C is suitable for secure device pairing (e.g., key exchange) and similar tasks. We also describe several use cases, as well as provide some performance data for a prototype implementation and a discussion of the security properties of L&C.*

1 Introduction and Motivation

The proliferation of many types of inexpensive personal devices, such as PDAs, cellphones, smart watches, and MP3 players, has been accompanied by the need to secure these devices and their communication with the "outside world." For example, natural applications include the need to securely connect one's personal device to an unfamiliar printer, fax machine, wireless projector, network access point, point-of-sale device, or another personal device. Of course, securely performing such connections is straightforward if we can rely on a global security infrastructure, such as a PKI, or a trusted online authority. Since such "heavy-weight" solutions are typically absent from heterogeneous personal device applications, we are interested in the more difficult problem of *peer device authentication*, which does not depend on a global security infrastructure or trusted online authorities. In particular, we are interested in applications where devices are physically present with each other but their communication channel is not visible or monitorable by the user(s). For example, we are interested in solutions to peer device authentication even in contexts when the communication channel is wireless (e.g., 802.11a/b/g, Bluetooth, or Infrared), which are particularly challenging, since such channels offer no physical evidence of a direct connection between devices[1].

While the more general problem of authenticating devices with no prior context remains difficult and partially unsolved, progress has been made in scenarios involving personal devices. Precisely because they are *personal*, the presence of the human user (owner) is assured and techniques have been proposed to engage the human user in the process of establishing secure communication. To this end, human-assisted authentication is a very timely and trendy research topic.

Relation to Seeing is Believing. One notable recent result is the *Seeing-is-Believing* (SiB) system proposed by McCune, et al. [15]. SiB takes advantage of the visual channel—since a human user is assumed capable of vi-

[1] Although we use wireless communication as a running motivation throughout this paper, the difficulty of peer device authentication is not confined to wireless links. For example, if the communication between the two devices is via wired Ethernet, a similar problem arises. Only a fully visible point-to-point physical connection would alleviate the peer authentication problem we study in this paper.

sually identifying the target device—to provide human-assisted authentication. This is done by requiring the human user to take a picture (with a personal camera-equipped mobile phone) of the 2D barcode affixed to, or displayed by, a target device and having the phone interpret the barcode and extract the cryptographic material identifying the device's public key. Meanwhile, the target device is supposed to communicate to the user's phone the (presumably) same cryptographic material via some wireless channel, e.g., Bluetooth. If the two versions of the cryptographic material match, the user's phone concludes that the target device's public key is authentic.

SiB provides a reasonable level of security, commensurate with what is realistic under the circumstances. Circumventing SiB would require the adversary to: (1) either hack into the target device and cause it to display wrong barcodes or physically plaster fake barcodes on the device, and, (2) mount a man-in-the-middle attack on the wireless channel. SiB is also quite practical particularly because it places very little burden upon the human user: visual identification of the target device and taking a picture of the barcode.

Loud and Clear Security. In this paper, we investigate an alternative and complementary approach—the use of the audio channel for human-assisted authentication of unfamiliar devices. We develop and evaluate a system called **Loud-and-Clear (L&C)**, which, like Seeing-is-Believing, places relatively little demand on the human user. In particular, L&C uses spoken natural language for human-assisted authentication; hence, it is suitable for secure device pairing (e.g., key exchange) and similar tasks, such as secure configuration.

The motivation for our work is three-fold:

1. While some mobile phones include a built-in photo (and even video) camera, many other types of personal devices are not similarly equipped. For example, a camera is not standard equipment on most PDAs (e.g., Treos, Blackberries, IPAQs and PalmPilots). It is also not present on digital music players and smart watches. Furthermore, even for mobile phones, an on-board camera results in a certain price differential, as compared to a similar camera-less phone.
2. The use of cameras is appropriate for most people, except for those who are visually impaired (e.g., legally blind). One possibility is to print barcodes in Braille, ask the visually-impaired user to identify the target device's barcode by touch and then take the picture (with the camera phone) at very close range. Albeit, the associated burden would be higher than in plain SiB.
3. Barcodes and cameras can be used in many normal everyday settings, such as offices, hotels and airports. However, there are two important underlying assumptions: (1) ample ambient light, and, (2) sufficient proximity between the two devices. In other words, in the presence of light-inhibiting environmental factors, such as darkness, smoke or heavy fog, SiB would not be applicable. Similarly, in a factory, lecture hall, athletic field, or battlefield scenario, it might be difficult for the human user to get close enough to the target device to take a meaningful picture of its barcode.

By relying on the use of spoken natural language to provide human-verifiable secure communication, L&C alleviates all of the above shortcomings of SiB. Moreover, the L&C system encodes authentication strings using auditorially-robust, grammatically-correct "MadLib" phrases, which allows human users to easily verify the authentication strings between peer devices. Indeed, we describe in this paper implementations of the L&C system applied to several peer device authentication scenarios. We also describe performance data for our implementations and security issues related to L&C. That is, we show that L&C provides an effective solution to many peer device authentication problems.

Nevertheless, the use of the audio channel for human-assisted authentication has its own drawbacks and limitations, which we readily admit from the outset:

1. Ambient noise is clearly an inhibiting factor for audio-based authentication. Whether comparing two audible sequences or comparing one such sequence to a displayed textual representation of the same sequence, noise makes authentication difficult. By the same token, the audio channel is not suitable for hearing-impaired human users.
2. As discussed later in this paper, L&C requires for at least one of the two devices to have a speaker (or an audio-out interface). The other device must either have a display or a speaker. While such interfaces are more common than cameras (as most personal devices are equipped with a speaker or a display, and often have both), we note that SiB does not require the use of a speaker or audio-out signal (it requires a camera/scanner and a display).
3. L&C places an alternate burden from SiB on the human user. In SiB, the user is asked to visually identify the target device and to take a picture of the device's barcode. In contrast, L&C requires the user to

either compare two audio sequences or compare one audio sequence to a displayed textual representation of the same sequence.

It is apparent from the above that, owing to their respective advantages and limitations, SiB and L&C nicely complement each other.

2 Related Work

There is a considerable amount of prior work on the general theme of establishing secure communication channels between devices with no prior trust relationships, in addition to the SiB work of McCune *et al.* [15], which we have already highlighted. Traditional PKI solutions involve rigid hierarchies and require the existence of trusted Certification Authorities (CAs) [18]. Unstructured certification techniques such as PGP [22] assume a certain small degree of separation among certified entities (i.e., a web of trust). An alternative is an online Trusted Third Party (TTP), as in Kerberos [27, 14]. However, third party solutions require constant presence and availability of an online trusted authority which is not realistic in our envisaged scenario. For this reason, the remainder of this section focuses on closely related prior research.

The well-known Diffie-Hellman key exchange protocol [33] allows two entities, with no prior secrets or secure association, to agree on a common secret key. However, precisely because no prior secrets are assumed, man-in-the-middle (MITM) attacks are possible [5]. A number of enhancements to Diffie-Hellman protocol have been developed. Bellovin and Merrit proposed the encrypted key exchange (EKE) protocol to prevent MITM attacks; but, EKE requires for both parties to pre-share a secret password [29]. Various EKE refinements [29, 30, 32, 21, 31] have been proposed, however, all of them still require a pre-shared secret password. This is clearly inapplicable in our target environment.

Secondary channels offer another means to defend against MITM attacks. A secondary channel can be used to verify that the keys computed at both devices are identical. A number of efforts have been made to involve a human user in the secondary channel in order to manually verify/compare keys (actually, hashes thereof). Unfortunately, since a a numerically-represent hash can be long and cumbersome, comparing hashes is an arduous process. To ease the burden, visual metaphors to represent hash values have been devised, including the use of fractal snowflakes [13, 24], random art [4] and flag representation [28]. These representations allow for human users to do rough similarity checking of strings, but they do not allow for fine-grain comparisons, as is required in security applications. Even the use of bar codes, as employed by SiB [15], are vulnerable to noise. Our use of grammatically-correct strings of words taken from phonetically-distant sets of words is more robust. That is, changing even a single bit in the authentication string (e.g., a cryptographic hash of a shared key) can be easily detected by the user in our L&C system.

A similar research direction to L&C is the textual representation of cryptographic strings. For example, the S/Key One-Time-Password [20] system represents an MD5 hash as a series of six short (up to four-letter) words. S/Key was designed for the automatic generation of pass-phrases, and the pass-phrases are not necessarily auditorially robust nor grammatically correct. That is, S/Key pass-phrases were not meant to be spoken, and there are some similar-sounding words in the S/Key word list. The word encoding for PGPfone [16], on the other hand, uses an auditorially robust word list. Still, it does not produce grammatically-correct string encodings, which makes it harder for human users to parse PGPfone phrases or even to read along with a spoken version of a PGPfone phrase. Our L&C system produces phrases that are grammatically correct and auditorially robust; hence, they are more suitable for human verification of phrase similarity.

Also of interest are schemes for constructing grammatically-correct pass-phrases from random passwords (e.g., see [6]). These approaches are designed to create ways of remembering passwords using the first letters of the words in a memorable pass-phrase. Thus, they are grammatically correct but not necessarily auditorially robust.

For peer device authentication, Stajano and Anderson proposed a method for establishing keys by means of a link created through physical contact [10]. However, due to the diversity in personal devices, it is impractical to expect all devices to have suitable physical connection interfaces. Likewise, it may also be infeasible to lug around connection interfaces and or interface converters for various devices. Balfanz, et al. [8] extended this approach to location-limited channels through the use of short-range wireless Infrared communication. Capkun, et al. [25] proposed a further extension to allow two previously unassociated devices to establish a key utilizing one-hop transitive trust.

3 Main Elements of L&C

In this section, we describe the main elements of the Loud-and-Clear (L&C) system.

As a first application, let us focus on the authentication of a target device to a personal device, assuming no prior context between the personal device and target device. We assume that in this, and most use scenarios, identification of the communicating devices is performed visually or tactilely by the human user.

Requirements. The specifics of target device authentication in L&C depend upon several factors, such as the type of authentication objects, directionality, the number of human users, and the device equipment. The following basic requirements are common to all use cases:

- There is at least one human user present with a personal device.
- At least one device has an audio interface, e.g. a speaker or a audio out plug (though, as discussed below, for the sake of completeness, L&C also supports the case of both devices having displays but no audio).
- The two devices must be able to communicate via some multiple-access broadcast medium, e.g., 802.11a/b/g, Bluetooth, Infrared, or wired Ethernet.

Figure 1 depicts some anticipated L&C use scenarios.

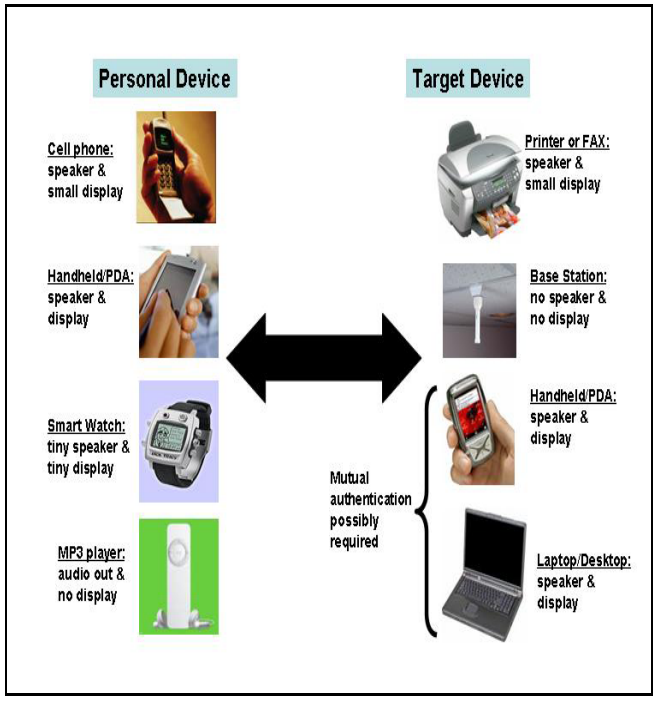

Figure 1. L&C Sample Use Scenarios

Classifying L&C Use Cases. Let us consider in more detail the factors that distinguish uses of L&C, including the type of authentication objects, directionality, the number of human users, and the device equipment. We begin by identifying two common types of *authentication objects*:

- **Target device's public key:** the personal device receives the target device's public key over a wireless channel (e.g., Infrared, 802.11a/b/g, Bluetooth) and an audio signal is used as a means of verifying this public key or the one-way hash of this public key. We call this a public authentication object.
- **Agreed-upon shared secret key:** the personal device and the target device compute a shared secret key via a wireless channel, whose one-way hash is then verified using an audio signal. We call this a secret authentication object.

A second factor that distinguishes use scenarios for L&C is the *directionality* of authentication, i.e., whether authentication is one-way (from target to personal device) or mutual (between the two devices). For example, in the former, the personal device may need to authenticate the target device's public key and, in the latter, each device may need to authenticate the other's public key or a shared secret authentication object.

A third factor is the number of human users. One classification has a single user with a personal device and the target device does not have an online human administrator. Another classification has two users, each with a personal device. In the former case, if mutual authentication is needed, the burden on the sole human user is essentially doubled.

More than anything else, the equipment available on each device influences the particulars of authentication. Some devices have both a display and a speaker, while others may have only one of these. Thus, some devices, such as low-end base stations, may at first appear unsuitable for L&C, since they lack both a speaker and a display. Nevertheless, they can be accommodated (in a manner similar to the way SiB handles display-less target devices), by affixing a sticker to the target device, which contains the L&C textual encoding of the target device's public authentication object, displayed in print and/or Braille form. In addition, any device, such as a point-of-sale device, with an embedded printer (or a printer itself) is easily supported by L&C, by having the device print an L&C encoding of the authentication object.

We consider four possible use types:

TYPE 1: hear and compare two audible sequences, one from each device.

TYPE 2: hear an audible sequence from the target device and compare it to text displayed by the personal device.

TYPE 3: hear an audible sequence from the personal device and compare it to text displayed by target device.

TYPE 4: compare text displayed by the personal device to text displayed by target device.

TYPE 1 is clearly the most taxing on the human user of the four; even if the audio sequences are short, comparing them is difficult due to different audio characteristics and the need for the human user to temporarily remember one sequence while waiting to hear the other. However, this requirement is unavoidable in some cases, as discussed below.

The TYPE 2 and TYPE 3 use cases are similar, but not the same, since the actual user actions are not identical for these two types. For example, listening to one's own device at very close range is always possible and convenient. Whereas, listening to an unfamiliar target device may require attuning to an unexpected volume, pitch, voice, and noise.

Concerning user requirement TYPE 4, it clearly does not involve any use of the audio channel. Nevertheless, we include it among our supported cases, since it may be used as an alternative or fall-back method if both devices only have displays (in which case, two human users could perform the role of the text-to-speech engine and manually perform the TYPE 2 or TYPE 3 use cases).

Of course, the equipment factor ultimately determines the amount of burden L&C places on the human user. Table 1 shows the types of user requirements corresponding to possible personal-target device combinations. Looking at rows 3 and 6, the choice between TYPE 3 or 1 and TYPE 2 or 1, respectively, can be dictated by the certain properties of the environment. For example, insufficient light, smoke or fog can make TYPE 1 the only viable choice. A visually-impaired user is also likely to choose TYPE 1 over TYPE 3 or 2, unless one of the two devices has a Braille display or sticker. Likewise, the TYPE 4 use case is infeasible for a visually-impaired user unless both devices have Braille displays (or a Braille sticker, in the case of the target device). In row 7, the choice between TYPE 3 or 4 is less clear. One deciding factor might be the comparative quality of the personal device's display and speaker.

		Personal Device		**Target Device**	
Row	Use Type	Display	Speaker	Display	Speaker
1	1	no	yes	no	yes
2	3	no	yes	yes	no
3	3 or 1	no	yes	yes	yes
4	2	yes	no	no	yes
5	4	yes	no	yes	no
6	2 or 1	yes	yes	no	yes
7	3 or 4	yes	yes	yes	no
8	1,2,3 or 4	yes	yes	yes	yes
9	n.a.	no	no	*	*
10	n.a.	*	*	no	no

Table 1. User Requirements for Various Device Combinations. The Use Type column indicates the allowed types of use cases, depending on the device characteristics indicated for the personal and target devices. We use '*' to denote a don't-care condition, we allow the 'Display' condition to include the ability to print or have an affixed sticker attached to device, and we allow the 'Speaker' condition to include any audio-out interface.

Vocalizable and Readable Representations. In L&C, the hash of the target device's public key (or agreed upon session key) must be verified by the user(s) of one or both devices. Comparing long (e.g., 160-bit) hashes is a tedious and cumbersome task for the average user. In order to make the process faster and less tedious, a hash must be represented in a more convenient form. In L&C, we represent the authentication object as a grammatically-correct text, with the expected use cases being situations (as in TYPE 2 or TYPE 3) where the user reads along with an audio text-to-speech reading of the text.

The text strings generated in L&C are based on the "MadLib" puzzles commonly used by children[2]. That is, we generate a sentence that is a grammatically correct—but usually non-sensical—from a string of bits.

This string of bits used in the L&C system will typically be the output from a one-way hash function. Our current implementation allows users to choose between SHA-1, MD2, or MD5 hash algorithms, which is then collapsed into smaller size using a technique similar to the "folding" that is done in the S/KEY One-Time Password System [20]. S/KEY partitions an MD5 hash into four 32-bit words, exclusive-ORs the first and third words as well as the second and fourth words. The resulting 64-bit quantity is then used to compute S/KEY phrase-words. L&C collapses the output to a smaller size by first dividing the hash into 10-bit sections. The sections are then paired off sequentially and XOR-ed. The

[2]In a MadLib puzzle, a funny story is created by having blanks in a text filled in with grammatically-appropriate words chosen by the player or his/her friend.

process is repeated for each pair until the desired level of entropy is attained. If less than half of the original entropy is needed, the process is repeated until the needed length is achieved. After the hash is collapsed to the desired level, the result is again divided into 10-bit sections. The number of 10-bit sections becomes the final length of the MadLib sentence. (For example, using SHA-1 with 80 bits of entropy would result in 8 MadLib keywords.)

Once the size of the binary input string is determined, an appropriately sized MadLib text is constructed. The text is generated from a template, which consists of a grammatical sentence (or group of sentences) with missing words, each of which being of various types, such as noun, adjective, adverb, verb, boy-name, girl-name, or animal. Each missing word is replaced with a word from a dictionary of appropriate words. The word replacing the MadLib keyword is determined by converting a 10-bit section of the collapsed hash into an integer and using that as the index into the internal dictionary database. For example, the following is a MadLib encoding produced by our prototype system, for encoding a 70-bit string (the filled-in words/word-phrases are shown in all caps):

> DONALD the FORTUNATE BLUE-JAY FRAUDULENTLY CRUSH-ed over the CREEPY ARCTIC-TERN.

Thus, in order to construct auditorially-robust text sequences, we need to produce a number of word lists of appropriate parts of speech, with the words in each list being as phonetically distant from each other as possible. Using metric for phonetic distance similar to that used by PGPfone [16], but restricted to words of the appropriate type, we can create auditorially-robust word lists for each of the word categories used in our MadLib sequences. In particular, we can do this as follows.

1. Construct a large set C of candidate words of the appropriate type. These should be common English words that can all be used in same place in a MadLib text sequence.
2. Select a random subset W of 2^k words from C, where k is the number of bits we wish to have this type of word represent (e.g., 8 bits for any noun).
3. Repeatedly find the closest pair (p, q) of words in W (using the phonetic distance metric) and replace q with a word from $C - W$ whose distance to any word in W is more than $d(p, q)$, if such a word exists. The resulting set will be a collection of phonetically well-spread words.
4. Order W so that each pair of consecutive words in W are as far from each other as reasonably possible. Doing this optimally is NP-hard, of course, but we can use a heuristic algorithm based on pairwise swapping of words in W to come up with a good order for W.
5. Assign integer values to the words in W according to a Gray Code, so that consecutive integers differ in exactly one bit but their respective code words are distant.

This algorithm helps to make the assignment of words to bit sequence auditorially robust—small changes (even to one bit) in the input should result in noticeably different sounding text strings.

Having described the use types for L&C and the way to produce MadLib text sequences from authentication objects, we describe some example use cases in more detail in the next few sections.

4 Bidirectional Authentication

Providing mutual authentication between devices that have no prior shared context is a challenging problem. Using L&C, the human participates in the procedure and the mutual authentication is established between the intended mobile devices. Using L&C in bidirectional authentication eliminates the need for a trusted party or any pre-shared secret.

The L&C protocol for mutual authentication enables Alice and Bob to establish a shared secret securely. First, Alice's device searches for nearby mobile devices that support the protocol. Alice then selects the device that she believes to be Bob's to connect.

Both Alice and Bob exchange their public keys and generate MadLib Sentences for them. Now both parties are ready to authenticate each other. Each party sends a signal to other device to read the MadLib sentence for its public key and compare it with the one that is computed from the received public key (using their respective displays or printouts). At this point, the Alice and Bob have mutually authenticated each other's device's public key. The authentication channel here is secure, since a human can sense from where the sound is coming from and any active attack is easily noticeable if the sentence read out is not same with the one shown on the screen. An attack is possible only if the attacker can find a second pre-image for the hash function used in MadLib creation, which is considered to be hard for any cryptographic Hash function.

Notice that the above scheme can be used in any public key protocol such as IKE [12], SSL/TLS [11]. However, a slightly modified scheme can also be useful if the agreed upon session key need to be authenticated. The only difference in this setting would be first doing

the Diffie-Hellman[33] key exchange and comparing the MadLib sentences generated from the agreed key in each device against each other. If the other device reads out the same sentence that is locally computed, then the session key could be considered authentic.

5 Unidirectional Authentication

L&C can be used for unidirectional authentication scenarios as well (TYPES 2 and 3).

In our first example, we consider a target device without a screen or a speaker, e.g., an 802.11 wireless access point. Any wireless device connected to the access point can read out the MadLib sentence generated from the received public key and the user can authenticate the access point by comparing the sentence read out in his machine with the one seen on the sticker attached to the access point.

Another common example is using a printer in a public place. Similar to the above, a printer could have a sticker with the sentence corresponding to its public key. However, printer is a more capable device of visualizing things on paper using its printing functionality. With the help of a button on a printer, it can be asked to print its MadLib sentence and the printer can be authenticated by the identical sentences that are heard form the computer and seen on the paper.

Authenticity of Internet sites is becoming more and more important as the Internet technology is being used in financial and privacy sensitive applications such as e-commerce, online banking or online health consultation sites. L&C can be used to authenticate Internet sites. In this setting, the authentication between user and the site is done in three rounds. First, the user provides its username and first level password to the website (e.g., using a cookie). Then the website plays the MadLib sentence associated with that username on user's web browser. If the the sentence is familiar, the one he always hears when logging on to this web-site, to the user, then he continues and enters the second level password to sign in to the site.

Sound is one of the main elements that enables visually impaired people to interact with the outer world. In this sense, L&C could be easily used to help visually impaired people to authenticate devices or even identities. One basic example could be the authentication scenario of bank ATM's. A visually impaired person could be given a MadLib sentence when she opens a bank account. This MadLib sentence is generated from the card number and its expiration date but using a keyed hash, with a secret key known only by the bank. When the card is inserted in an ATM, the ATM will generate and read the MadLib sentence corresponding to the account number.

6 "Presence" Confirmation

Similar to Seeing-is-Believing [15] Loud and Clear security can also be used to prove the "presence" of a separate device. This "presence" property occurs when a device containing a speaker or display, without the assistance of a human, needs to confirm the presence of some other device within a given range.

To detect the presence of a nearby device, the device containing the speaker or display begins by generating a nonce value, hashes the nonce value, and converts the hash into the correct MadLib. The device then either displays or automatically speaks out the MadLib, noting the time when the MadLib was displayed or spoken. Anyone who is nearby the device will be able to read or hear the MadLib and can enter the MadLib sentence into their personal device. The personal device can then simply reverse the MadLib encoding process by determining the index of the word in its corresponding dictionary and using the bits of the index as bits of the hash. Once the entire hash has been reverse-computed from the MadLib then the personal device can broadcast the hash of the nonce over the wireless channel. When the original device receives the hash of the nonce over the wireless channel, the device knows that someone has been present at the display's line-of-sight or within hearing range of the speaker.

While the "presence" property can be achieved by devices, the "presence" property is not robust. It is impossible for the device to determine how many devices are within hearing range or how many are capable of seeing its display. The only thing the device is capable of is verifying that the hash received over the wireless channel is the same as the one generated. If desired, it is possible to determine the time difference between when the nonce hash is received over the wireless channel and when it was first displayed or spoken.

Even though the "presence" property is not robust, it is not without its own uses. A device can use the "presence" property to limit the authority to control the device to users located within view or hearing range of the same device. This property can also be used as means of triggering the power saving feature of a device. Before a device switches to a lower power consumption state, the device displays or speaks the MadLib. If no response is received over the wireless channel after a certain time period, then the device determines that no one is within range and can safely switch state.

7 Implementation and Performance

In this section we describe the prototype implementation of L&Cand evaluate its (experimental) performance taking into account various cost factors.

PC and Pocket PC Application . Since L&C is intended for a variety of mobile computing platforms, portability is one of the main requirements. We built L&C using the highly portable *Ewe* Programming System [7], which allows the development of Java applications.[3] The implementation runs on any Pocket PC (iPAQ in our experiments) and any Windows PC or laptop.

At bootstrap time, the system prompts the user to select between DH and RSA key exchange. One of the participants/devices (say, Bob) initializes L&C and waits for a TCP connection on a well-known port over either the 802.11 or Infrared channel. The other participant (Alice) physically approaches Bob (this is particularly relevant for Infrared), initializes its L&C application and connects to Bob. Then, the two participants exchange either DH parameters and respective public keys or RSA public keys. L&C converts the hash of the received public key into a MadLib sentence (as discussed above), displays the sentence and offers the user the option to vocalize the sentence.

On the iPAQ, the user can sound out the MadLib sentence by simply pressing a button. If there are two users, they compare what they see or hear (or both) and verify the received public keys. At this stage, any MITM attack can be detected and the exchange cancelled before the devices perform any further computations.

In the DH case, the two L&C -enabled devices compute the shared secret $g^{ab} \pmod p$. In the RSA case, the two devices use each other's public keys to encrypt their contribution to the session key (a random quantity Z_n). They send the encrypted partial secrets over the wireless channel, decrypt the received secrets and XOR them to obtain the shared secret. The protocol can use newly generated or predetermined DH parameters and RSA public keys. Since the generation of DH parameters (p, g) and RSA modulus (n) is very time-consuming, L&C is configured to use pre-set parameters. However, the DH key pair as well as the partial secret in the RSA key exchange are always generated in real time using a cryptographically strong pseudo-random number generator function in the Ewe security package.

[3] Ewe is currently available for the following platforms: PocketPC (Windows CE), MS SmartPhone, Casio BE-300, HandHeldPC Pro, Sharp Zaurus, Linux PC, Windows PC and any Java 1.2 VM.

Alternatively, L&C can be used to generate and verify the MadLibs of the shared secret. This mode equally secure, however, it has the disadvantage that, if an active attack were to occur, it would be detected later, after the devices perform expensive modular exponentiations.

L&C can be used by a user-attended personal device to verify the public key of an un-attended target device. This corresponds to uni-directional authentication described above. Alice (personal device) initiates a connection and starts the key exchange. Once the public keys are exchanged, Alice can listen to Bob's (target device) public key MadLib vocalized (or displayed) by her device and compare to the MadLib vocalized (or displayed) by Bob. Note that Alice does not need to physically cause Bob to vocalize its MadLib, e.g., by pressing a button on Bob's keyboard. Instead, Alice uses her own device to send a message to Bob, which prompts it to begin the vocalization.

Implementation Details . Ewe does not provide many low-level cryptographic primitives. Therefore, to implement DH- and RSA-based key exchange protocols, we added a lightweight cryptographic API to Ewe's Java libraries. For this purpose, we ported the *Bouncy Castle* crypto package [1] for JDK 1.3. The RSA implementation uses the Chinese Remainder Theorem (CRT) to speed up decryption operations. For hashing we used Ewe's built-in SHA-1.

Owing to modular software design, L&C can utilize a variety of Text-to-Speech (TTS) engines. However, most C/C++ speech engines are platform-dependent, while those written for mobile devices are mostly proprietary. Furthermore, Java-based TTS engines are available for specific JVM-s that are unsuitable for resource-constrained devices, such as smartphones and iPAQ-s. Specifically, Sun offers JSAPI and FreeTTS Java TTS engine implementations. However these run only on Java 1.4. Therefore, we employed existing TTS applications that could be used by L&C—*Digit for PC* and *Pocket PC* by Digalo (www.digalo.com), which is a simple lightweight clipboard reader that uses the *Elan Speech Engine*. Our application copies the text to-be-vocalized onto the system clipboard and Digit speaks it out automatically or when the user presses a button on the application window. Digit is both initialized and terminated from within the Ewe program.

Both FreeTTS and Bouncy Castle crypto package are written solely in Java and do not link to native platform-specific libraries, facilitating L&C's platform independence. At present, we tested L&C on Pocket PC, Win-

dows PC and Linux PC. L&C can also be used with the rest of the aforementioned platforms by changing the TTS engine. We are currently working on porting Sun's FreeTTS and JSAPI into Ewe.

L&C Performance . In this section we evaluate L&C performance using a commodity laptop PC as a target device and a low-end iPAQ as a personal device. The laptop PC was equipped with: Intel Pentium M Centrino 1.7GHz, 400MHz FSB, 2MB Cache and 512 MB RAM running Windows XP. An iPAQ was a Compaq 3650 equipped with: an Intel Strong ARM SA-1110 32-bit RISC processor operating at 206MHZ, with 31.25 MB RAM, 16 MB ROM running Windows CE version 3.0.9348 (PocketPC 2002). For the 802.11 channel we configured a wireless subnet consisting of: one wireless router, two iPAQs and a PC. The channel's nominal bandwidth for all devices was 11 Mbps. The Infrared ports of all devices operated at 115 Kbps.

We evaluated L&C for both DH- and RSA-based key exchange. It is generally accepted that, for any b, factoring b-bit integers requires about the same amount of time as computing discrete logarithms in $(b - x)$-bit fields, where x is a small constant, roughly around 20 [17]. Therefore, we used 1024-bit moduli for both RSA and DH.

Table 2 lists processing times for each operation that is part of a bi-directional voice-only (TYPE 1) L&C session. Measurements for operations 1, 2, 13, 15, 25 and 27 were obtained as the average over 20 L&C sessions operated by human users. The rest were obtained over over 300 bulk repetitions of L&C sessions. We used Ewe's timing function, which offers (only) 10 ms precision. L&C initialization times (row 1) are obtained after RAM has been reset, so that they include the time to load all the application (Ewe Vm, L&C class files and Digit) in memory.

In the experiments we used syntactically correct MadLib sentences consisting of 10 words, 7 of which are S/KEY-generated. The sentence format is:

> *"Alice: My public key is BOYNAME the ADJECTIVE ANIMAL ADVERB VERB-ed over the ADJECTIVE ANIMAL."*

In reality, the time Alice spends on connection setup is comprised of: (1) time to enter the target device's network address (IP address or "infra-red"), press "Enter" and "Connect" buttons and, lastly, align the devices if the Infrared channel is used. It also includes the time for the application to reach the *accept()* or *connect()* calls.

No	Operation	iPAQ	PC	#
1	Vm, and GUI initialization	2430	120	0
2	Digit initialization	18310	1092	1
3	Connection setup by user	1502	910	1
4	TCP Connection est., 802.11	3.2	0.4	1
5	TCP Connection est., IR	3.4	-	1
	DH-based Protocol			
6	key-pair gen.	3384.7	78.2	1
7	pub. key transmission, 802.11	5.6	0.1	1
8	pub. key transmission, IR	6.1	-	1
9	param. transmission, 802.11	10.1	0.2	1
10	param. transmission, IR	11.4	-	1
11	shared secret computation	3540.2	268.9	1
12	pub. key MadLib generation	626.9	30.6	2
13	pub. key MadLib vocalization	9144	6360	4
14	shared secret MadLib generation	362.7	17.1	0
15	shared secret MadLib vocalization	9090	6334	0
	TOTAL TIME	64,603.1	28,123.1	
	RSA-based Protocol			
16	pub. key transmission, 802.11	11.6	0.6	1
17	pub. key transmission, IR	14.2	-	1
18	partial secret gen.	52.0	5.1	1
19	partial secret encr.	14.3	1.2	1
20	partial secret transmission, 802.11	6.7	0.2	1
21	partial secret transmission, IR	11.4	-	1
22	partial secret decr.	2422.4	143.9	1
23	shared secret computation	1.1	0.1	1
24	pub. key MadLib generation	376.4	17.9	2
25	pub. key MadLib vocalization	9200	6355	4
26	shared secret MadLib generation	361.4	16.2	0
27	shared secret MadLib vocalization	9151	6375	0
	TOTAL TIME	59,876.1	27,609.1	

Table 2. Processing times (in ms) for components of a bi-directional voice-only (TYPE 1) key agreement session.

For the experiments, the Infrared ports were pre-aligned and default network addresses were used. Hence, Alice needed only to press two buttons to initiate the connection.

We examine in detail the scenario involving two users performing bi-directional voice-only of each other's public keys and the computed session key. Each user verifies the other's public key (or shared secret) after hearing the corresponding MadLib spoken by the remote device and personal device, in either order.[4] This means that our scenario includes four MadLib generations: two per device. It also includes four MadLib vocalizations, which must

[4]Our preliminary user studies indicate that MadLibs should not be spoken simultaneously, else users have difficulty understanding the sentences. They also suggest that users are able to remember 5-10 word sentences for short periods of time.

take place sequentially. The total time is approximately the sum of the time taken for the above operations as well as the cryptographic operations, plus, any delays introduced by the users. The last column in Table 2 shows the number of individual operations involved.

As can be seen from Table 2, a DH-based L&C session (where the public keys are vocalized by both devices) can complete in approximately 65 seconds on the iPAQ and 29 seconds on the PC. An RSA-based L&C session can be completed in approximately 60 seconds on the iPAQ and 28 seconds on the PC. The following operations are the most time-consuming: (1) DH shared secret computation, (2) DH key-pair generation, (3) RSA partial secret generation, (4) RSA partial secret decryption and (6) MadLib vocalization. We observe that communication costs represent only a small fraction of the total cost.

We emphasize that **"TOTAL TIME"** reflected in Table 2 is the absolute **WORST CASE** scenario representing the heaviest possible use of all components of L&C. In contrast, the simplest (and probably the most common) anticipated use scenario corresponds to TYPE 2 of Section 3. Recall that TYPE 2 involves uni-direction authentication of the target device's public key. In it, the human user (iPAQ owner in our experimental setting) receives the hash of the target device's (PC) public key, reads the corresponding MadLib sentence from the iPAQ's display and compares it to that vocalized by the PC. In this minimal case, "TOTAL TIME" for the iPAQ with the DH protocol would include the following table elements (assuming 802.11): 2.iPAQ, 3.iPAQ, 4.iPAQ, 7.PC, 9.PC, 12.iPAQ, and 13.PC. This amounts to roughly $26,800$ ms, a far cry from $64,603$ ms in table. Whereas, "TOTAL TIME" for the iPAQ with the RSA protocol would include the following (again, assuming 802.11): 2.iPAQ, 3.iPAQ, 4.iPAQ, 16.PC, 24.iPAQ and 25.PC. This totals roughly $26,543$ ms, as opposed to $59,876$ reflected in the table.

Naturally, the time to speak out a MadLib sentence is proportional to the word length of the sentence as well as to the number of syllables in each S/KEY-generated word. Therefore, it can vary for the same word-length sentences. Since 4-5 S/KEY-generated words already provide sufficient security, this time can be reduced.

In the table, the last column indicates the number of times the corresponding operation needs to be added for the total time calculation of a bidirectional authentication session, during which all MadLibs are vocalized. The sessions are either Diffie-Hellman or RSA-based key exchange. We use 1024-bit DH and RSA keys. DH is used with pre-set parameters (p, g) and CRT-aided RSA with parameters (n, e, d_p, d_q). For each session, operations take place in the order they are listed. The Virtual Machine, GUI and Digit initialization time as well as the user connection setup time is the same for both agreement protocols. In DH key agreement, the shared secret computation is an exponentiation modulo p (where p is a 1024-bit prime), and, in RSA key agreement, the secret computation is a simple *exclusive or* operation over the partial (decrypted) secrets. The DH private key and the RSA partial secret are 1022 to 1023-bit pseudo-random numbers. Connection establishment time is defined as the time taken by the TCP socket *connect()* system call to connect to the accepting process running on the iPAQ or the PC. The connecting process always runs on the iPAQ. We measured L&C sessions over the IR channel only between iPAQs.

As expected, the processing times observed for iPAQ's is substantially higher than those for Windows PC. In particular, parameter transmission and shared secret computation is an order of magnitude cheaper on a PC. DH key pair and RSA random partial secret generation has 40-50 and 10-15 times respectively longer running time on iPAQ. Similarly, RSA encryption and decryption is 15-20 times less expensive on PC. Also, MadLib generation is approximately 20 times more expensive on PocketPC. [5]

For RSA, the most expensive operation is the decryption of the other party's partial secret, while, in DH, it is the computation of the shared secret. By comparing operations 11 and 19, we observe that RSA is approximately 40% less expensive than DH in terms of shared secret generation. This is because RSA decryption using the CRT method involves exponentiations modulo p and q, where both are 512-bit values. On the other hand, DH secret computation involves exponentiations modulo p, where p is a 1024-bit prime. On the other hand, DH incurs high computational cost for key-pair generation, since each device must perform a modular exponentiation in order to compute the public key.

Since packet traversal of the protocol stack – and not the actual transmission over the physical medium – is the dominating factor of the communication cost, the difference in communication costs over 802.11 and Infrared channels is insignificant. It is also evident that the time for a typical user to set up a connection is greater on an

[5] We believe that the observed difference between the running times of public and shared key MadLib generation is due to the fact that the Java classes that are used for the generation (MadLib, ewe.security.SHA1) are already loaded when the shared key MadLib is generated.

iPAQ than on a PC. This is due to less-than-user-friendly GUI and slower rendering of the GUI components on the Pocket PC.

In summary, processing and memory limitations on the iPAQ result in significantly longer delays than on the PC. (However, it is important to stress that we used **low-end** iPAQs in our experiments. Using current state-of-the-art iPAQs or other similar devices would greatly reduce delay.) Furthermore, RSA-based key agreement requires marginally less time than its DH-based counterpart. Experimental results suggest that the most plausible way to reduce the protocol time is to shorten MadLib sentences and speed up the TTS engine initialization time. We conclude that L&C is a viable solution on platforms with moderate computation and communication capabilities[6].

8 Security Analysis

Assuming that all underlying cryptographic primitives, cryptographic hash functions and public key algorithms (RSA and Diffie-Hellman key agreement) are secure, the security of L&C depends on the adversary's inability to: (1) interfere with the audio or visual channel and (2) compromise the target device. In this section, we discuss the security of the cryptographic primitives and compare the security of the alternative channels. We then discuss some concrete attack scenarios.

Cryptographic Primitives . To establish a secure bidirectional channel between communicating devices (as described above), L&C uses either the ephemeral DH key agreement or RSA-based mutual key agreement. The security of both methods is based on well-known and believed-to-be-hard computational problems. Both types of key agreement can be protected against man-in-the-middle (MITM) attacks by verifying the exchanged public keys. As mentioned in Section 7, security against MITM attacks can be achieved by using L&C to verify the agreed-upon shared secret. However, public key verification is preferable because the users can detect the attack early in the process and abandon it without performing further expensive computations.

In the DH case, Alice and Bob exchange their respective public keys $y_a = g^{a_a} \bmod p$ and $y_b = g^{a_b} \bmod p$. a_a and a_b are the corresponding secret keys known only to Alice and Bob, respectively. By verifying that the received DH public key y originates with the intended sender, Alice (Bob) can be certain that the session key she computes as: $y_b^{a_a} \bmod p$ (or $y_a^{a_b}$) is the same for the two parties. As commonly done, we assume that non-specific parameters p and g are both system-wide and long-term. However, the device-specific key pairs (g^a, a) can be ephemeral since their generation is not particularly expensive (as discussed in the previous section).

In case of RSA, Alice and Bob also exchange their respective public keys (n_a, e_a) and (n_b, e_b). Alice encrypts a random number $s_a \in_R Z_{n_b}$ as $s'_a = s_a^{e_b} \bmod n_b$ and sends it to Bob. s_a is Alice's contribution to the session key, also referred to as partial secret. Bob decrypts it: $s_a = (s'_a)^{d_b} \bmod n_b$, where d_b is Bob's RSA secret key. Both parties perform this procedure and combine their contributions to compute the session key: $s = s_a \oplus s_b$. If Alice and Bob each verify, using L&C the other party's public key each can be certain that the encrypted session key contributions cannot be decrypted by an adversary and the final session key is secure. Even if the adversary interferes with the transmission of the encrypted secrets, the result would be failed key agreement which can be easily detected by both parties.

In L&C, the hash function of the transmitted public key (or the agreed-upon shared key) is converted to a MadLib sentence. During the conversion, the length of the MadLib sentence can be set to the desired entropy level. However, if the adversary can find a second preimage of the hash value that encoded in the MadLib sentence, then it can attack the communication channel and replace the transmitted key with the collided pre-image, causing Alice and Bob to falsely verify the key agreement. However, second preimage resistance is a required property for cryptographic hash functions and we assume that it holds.

Although the length of the MadLib sentence can be set to achieve the desired level of security, it is commensurate with the burden of comparing longer sentences. In many applications, the user is not expected to memorize the sentence, but a longer sentence still results in a longer verification time. The danger of using short sentences becomes real when the long-term public keys are used. In this case, the adversary may have enough time to find a collision via a brute force attack. To prevent such attacks, we prefer using DH-based over RSA-based key agreement. According to our experimental results, the DH key exchange portion of L&C takes only a few seconds even in resource-constrained PDAs. Finding a collision even for a very short sentence (e.g., 6-7 words) in a few seconds is still infeasible with today's technology, since it would require 2^{59}–2^{69} operations. Once a session key

[6] L&C source code, installation instructions for Windows and Pocket PC as well as a video demonstrating L&C use-cases can be found at http://www.ics.uci.edu/ccsp/lac.

is established, finding the collision on the MadLib sentences does not give the adversary any advantage.

Alternative Channels . Authentication of communicating parties without the assistance of a trusted party requires a communication channel that is secure against active attacks. In this section, we discuss several alternative channels and their security compared to that of L&C.

Wired channels between two devices, such as cross-over Ethernet, direct USB or serial connection, can be considered secure against active attacks since both ends of the cable are directly connected to the intended devices and the entire cable is visible to the human user. However, lugging around different types of wires in order to to communicate with different types of devices is quite cumbersome.

Wireless alternatives, such as 802.11a/b/g, Infrared and Bluetooth, are not secure against MITM attacks. A human user is essentially incapable of identifying which two devices are actually communicating over a wireless channel. The only way to be sure that the intended devices are communication without interference is through status indicators on devices, which is prone to errors and attacks. Among wireless alternatives, Infrared is considered most secure since it requires a direct line-of-sight between devices (which can be visually inspected). However, it is still possible for a well-placed adversary in a device-crowded setting to interfere with communication, since the human user cannot determine the exact direction of an infrared signal.

As discussed in Section 1, the visual channel – exemplified by SiB is a very viable alternative. However, many devices lack cameras as well as large-enough displays and powerful-enough CPUs to perform necessary image processing (e.g., to extract the barcode). Furthermore, environmental factors inhibiting the use of cameras as well as sight-impaired users limit the applicability of the visual channel. Other hash visualization techniques [19, 23, 9, 2] are also viable alternatives for verifying public and/or session keys. They can detect MITM attacks as long as the comparison is done very carefully and the environment is well-illuminated.

Attacks Against L&C. In L&C, the man in the middle adversary is easy to notice. When Alice asks for Bob's device to read its MadLib sentence loud, even if Alice is far enough to be tricked by an adversary device close to Bob, Bob can easily say if it is his device talking by just checking the screen or sensing the origin of the sound. Furthermore, Bob and Alice can detect the active attack if the adversary generates sound without them asking the other device to do so.

An adversary may attempt to attack L&C by using malicious software surreptitiously installed in the target device. Malicious software might tamper unnoticed with the input and output of the desired application. This problem is sometimes referred to as *secure windowing* or *establishment of a trusted path to system window* [26]. Such attacks can be prevented by using a good system window policy, e.g., displaying application windows in predetermined locations or, as proposed in [3], using predetermined window backgrounds known only to the user. A malicious application may also try to process the MadLib sentence from screen pixels and redisplay it, which would appear legitimate to the user. This can be prevented by limiting the screenshot and copy operations to the application that is on top at the time. EROS, a trusted window system, proposed by Shapiro, et al. [26] provides further details on how to achieve this type of protection.

An adversary may also emit interference in the form of ambient noise in order to alter L&C audio. However, this attack is easy to detect since two different sound sources would be involved. Moreover, the adversary would need near-perfect synchronization, a-priori knowledge of the MadLib sentence and plenty of computational power to be able to produce correct sounds that will make the sentence sound like a MadLib sentence corresponding to some public key picked by the adversary.

Excessive ambient noise could also prevent the user(s) from hearing each other's devices, but the source of the noise would be obvious. Even if it cannot be avoided, ambient noise would at best result in denial of service. In such cases, user(s) can fall back to comparing MadLib sentences displayed or printed on the screens (or stickers) of the respective devices.

9 Conclusion

In this work, we developed the **Loud-and-Clear (L&C)** system for human-assisted device authentication. L&C places relatively little burden on the human user, since it is based on the audio channel and uses a text-to-speech engine to read an auditorially-robust, grammatically-correct sequence derived from an authentication string (either a public key or a secret session key). We also described some (anticipated) common use cases and provided experimental performance data for a prototype implementation.

References

[1] Bouncy Castle Crypto APIs. Available at: http://www.bouncycastle.org/index.html.

[2] Visual key fingerprint code., 1996.

[3] *The battle against phishing: Dynamic Security Skins*. ACM Press, 2005.

[4] Adrian Perrig and Dawn Song. Hash visualization: A new technique to improve real-world security. In *Proceedings of the 1999 International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC '99)*, pages 131–138, July 1999.

[5] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press Series on Discrete Mathematics and its Applications. CRC Press, 1997.

[6] M. J. Atallah, C. J. McDonough, V. Raskin, and S. Nirenburg. Natural language processing for information assurance and security: An overview and implementations. In *Proceedings of New Security Paradigm Workshop*, pages 51–65, 2000.

[7] Michael Brereton. Ewe java vm for pocketpc. Available at http://www.ewesoft.com/.

[8] Dirk Balfanz, D.K. Smetters, Paul Stewart, and H. Chi Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Symposium on Network and Distributed Systems Security (NDSS '02)*, February 2002.

[9] Carl Ellison and Steve Dohrmann. Public-key support for group collaboration. *ACM Trans. Inf. Syst. Secur.*, 6(4):547–565, 2003.

[10] Frank Stajano and Ross Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Security Protocols, 7th International Workshop*, 1999.

[11] A. Freier, P. Karlton, and P. Kocher. The ssl protocol: Version 3.0, 1996.

[12] D. Harkins and D. Carrel. Rfc2409: The internet key exchange(ike), 1998.

[13] Ian Goldberg. Visual key fingerprint code., 1996. Available at http://www.cs.berkeley.edu/iang/visprint.c.

[14] Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An authentication service for open network systems., March 1998. Manuscript.

[15] Jonathan M. McCune, Adrian Perrig, Michael K. Reiter. Seeing-Is-Believing: Using Camera Phones for Human-Verifiable Authentication. In *2005 IEEE Symposium on Security and Privacy*, pages pp. 110–124, 2005.

[16] Patrick Juola and Philip Zimmermann. Whole-word phonetic distances and the pgpfone alphabet. In *Proceedings of the Fourth International Conference on Spoken Language Processing (ICSLP)*, volume 1, 1996. http://www.asel.udel.edu/icslp/cdrom/vol1/005/a005.pdf.

[17] Arjen K. Lenstra and Eric R. Verheul. Selecting cryptographic key sizes. *Journal of Cryptology: the journal of the International Association for Cryptologic Research*, 14(4):255–293, 2001.

[18] Loren M. Kohnfelder. Towards a practical public-key cryptosystem., 1978. B.sc thesis, MIT Department of Electrical Engineering.

[19] Philip D. MacKenzie, Sarvar Patel, and Ram Swaminathan. Password-authenticated key exchange based on rsa. In *ASIACRYPT '00: Proceedings of the 6th International Conference on the Theory and Application of Cryptology and Information Security*, pages 599–613, London, UK, 2000. Springer-Verlag.

[20] N. Haller. Available at RFC1760: The S/KEY One-Time Password System, February 1995.

[21] P. MacKenzie, S. Patel, and R. Swaminathan. Password authenticated key exchange based on RSA. In *Advances in Cryptology – ASIACRYPT*, pages 599–613, 2000.

[22] P. Zimmermann. *The Official PGP User's Guide*. MIT Press, 1995.

[23] Adrian Perrig and Dawn Song. Hash visualization: A new technique to improve real-world security. In *Proceedings of the 1999 International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC '99)*, pages 131–138, July 1999.

[24] Ralph Levien. PGP snowflake, 1996. Source code available at: http://packages.debian.org/testing/graphics/snowflake.html.

[25] S. Capkun, J. Hubaux, and L. Buttyan. Mobility helps security in ad hoc networks. In *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing(MobiHoc 2003)*, June 2003.

[26] Jonathan S. Shapiro, John Vanderburgh, Eric Northup, and David Chizmadia. Design of the eros trusted window system. In *USENIX Security Symposium*, pages 165–178. USENIX, 2004.

[27] S.P. Miller, C. Neuman, J.I. Schiller, and J.H. Saltzer. Kerberos authentication and authorization system. In *Project Athena Technical Plan*, page pg section E.2.1, 1987.

[28] Steve Dohrmann and Carl Ellison. Public key support for collaborative groups. In *Proceedings of the First Annual PKI Research Workshop*, April 2002.

[29] Steven Bellovin and Michael Merrit. Augmented encrypted key exchange: a password-based protocol secure against dictionary attacks and password file compromise. In *First ACM Conference on Computer and Communications Security CCS-1*, pages 244–250, 1993.

[30] Steven M. Bellovin and Michael Merrit. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *IEEE Symposium on Security and Privacy*, pages 72–84, 1992.

[31] T. Wu. The secure remote password protocol. In *Network and Distributed System Security Symposium*, February 1999.

[32] V. Boyko, P. MacKenzie, and S. Patel. Provably secure password authentication and key exchange using diffie-hellman. *Advances in Cryptology–EUROCRYPT*, 1807 of Lecture Notes in Computer Science:156–171, 2000.

[33] W. Diffie and M. E. Hellman. New directions in cryptography. In *IEEE Transactions on Information Theory*, pages IT–22(6):644–654, November 1976.