Seeing-Is-Believing:
Using Camera Phones for Human-Verifiable Authentication

Jonathan M. McCune, Adrian Perrig, Michael K. Reiter

November 2004
CMU-CS-04-174

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Today it is difficult to conveniently provide strong authentication between devices that share no prior context, without the assistance of a trusted authority. We present, analyze, and propose applications for Seeing-Is-Believing, a system that utilizes machine vision on camera-phones to achieve security properties formerly unattainable in a non-intrusive way.

We gratefully acknowledge support from the Army Research Office contract number DAAD19-02-1-0389 entitled “Perpetually Available and Secure Information Systems”.

Keywords: authentication, key distribution, visual channel, camera, barcode, mobile phone

1 Introduction

It is an open problem today to configure security properties between devices in ways that are easily understandable by non-expert users. Current work relies on communications channels that are not directly human-verifiable. This is best exemplified by the problem a user faces when he wants to securely connect his wireless device to *that* other device (e.g., a network printer, an 802.11 base station, or another wireless device). In general, it is exceedingly difficult to determine which device is at the other end of a wireless connection without a side-channel or other out-of-band knowledge. Balfanz et al. describe this as the problem of achieving *demonstrative identification* [4]. We approach this problem with the premise that, in many situations, a user can identify the desired device visually.

As camera-equipped mobile phones rapidly approach ubiquity, a platform for secure human-verifiable electronic communication becomes available. Today's mobile phones increasingly have Internet access and come equipped with cameras, high-quality displays, and short-range Bluetooth wireless radios. They are powerful enough to perform public key cryptographic operations in under one second. This constitutes a unique and powerful platform for security applications that can be deployed quickly and easily to millions of users.

In this paper, we propose to use the camera on a mobile phone as a new *visual* channel to achieve security properties formerly unattainable in an intuitive way. We term this collection of approaches Seeing-Is-Believing (SiB). Recent improvements in the image quality and processing power attainable on camera-phones has enabled the development of effective barcode-reading software [1, 26, 32, 41]. In SiB, one device uses its camera to take a snapshot of a barcode encoding cryptographic material. This barcode can contain a commitment to the creating device's public key material, or an array of barcodes can be used to send key material directly. Barcodes can be pre-configured and printed on labels attached to devices, or they can be generated on-demand and shown on a device's display.

After surveying related work in Section 2, provide an overview of SiB in Section 3. We then consider the application of SiB to several problems in computer security, including authenticated key exchange between mobile devices (Section 4); demonstrative identification of, and secure connection to, a particular wireless device (Section 5); and secure verification that the desired application currently owns the display of a TPM-equipped[1] computer (Section 6). We also show how this technology can be used to achieve slightly weaker—but still quite valuable—security properties in the context of, e.g., a smart home (Section 7). Our implementation is detailed in Section 8, with a security analysis is Section 9. Section 10 concludes.

2 Related Work

We first review related work on authentication involving mobile devices, then consider barcode scanning with camera phones.

[1] The Trusted Platform Module (TPM) is the physical realization of a trusted computing specification released by the Trusted Computing Group (TCG) [2, 3].

2.1 Authentication

In this paper, we study authentication between two co-located entities with no prior trust relationships. Since using a public key infrastructures relies on the existence of trusted certifiers (e.g., hierarchical certification [22] or unstructured certification [44]), we do not consider these approaches here. Similarly, trusted third party approaches, such as Kerberos [29, 36], assume an online trusted authority which may not exist in our setting. We now review closely related research in chronological and topic-related order.

A common mechanism to establish a secure channel between two entities is to use Diffie-Hellman key establishment [11]. Unfortunately, a man-in-the-middle (MITM) attack is possible if the two entities do not share any established trusted information [28]. Bellovin and Merrit propose the encrypted key exchange (EKE) protocol, which prevents the MITM attack if both parties share a secret password [6]. Many researchers have refined this approach [6, 7, 8, 25, 42], but they all require a shared secret password between the two entities, which may be cumbersome to establish in many mobile settings.

Another approach to defeat the MITM attack is to use a secondary channel to verify that the keys at both entities are identical. An approach that several researchers have considered is that a human can manually verify that the generated keys are identical. Since comparing the hashes of the two keys is cumbersome, researchers have devised visual metaphors that represent the hash to make it easier for people to perform the comparison: Levien and Goldberg have devised the Snowflake mechanism [15, 24], Perrig and Song have proposed to use Random Art as a visual hash [30], and Dohrmann and Ellison have proposed a colorful flag representing a hash of the key [12].

To set up secure trusted information, Stajano and Anderson propose to set up keys through a link that is created through physical contact [35]. However, in many settings, devices may not have interfaces that connect for this purpose, or they may be too bulky to carry around. Balfanz et al. extend this approach to use location-limited channels, by using short-range wireless infrared communication [4]. Their approach is the most closely related work and we discuss it in more detail in a later section. Čapkun, Hubaux, and Buttyán have further extended this research direction [40]. They make use of one-hop transitive trust to enable two nodes that have never met to establish a key.

Researchers have studied the problem where a user wants assurance that what he sees on the screen of a computer is really the content sent from a trusted provider or displayed by a trusted application. Tygar and Whitten consider trojan horse web applications that may be able to monitor keystrokes and steal private application [39]. They combat this problem by window personalization. This is best described as a heuristic and does not provide cryptographic security guarantees.

Felten considers "web spoofing" where users are fooled into thinking a malicious web page is coming from a trusted provider [13]. Ye and Smith address the web spoofing problem and propose *trusted paths* from web browsers to users. Their solution is based on window borders that flash in a particular way, such that malicious content or a malicious application is unable to easily match the correct pattern. This solution provides no cryptographic guarantees, and we believe it requires excessive user diligence.

2.2 Camera Phones

We now discuss related work in using camera-equipped mobile phones to recognize bar codes. Several projects exist that seek to allow camera-equipped mobile phones to interact with physical objects through the use of 2D barcodes. Rohs and Gfeller developed their own 2D code explicitly for use with mobile phones, emphasizing their ability to be read from electronic screens and printed paper [32]. Woodside developed *semacodes*, which is an implementation of the Data Matrix barcode standard for mobile phones [9, 41]. Woodside considers the primary application of semacodes as containers for a URL which contains information about the physical location where the barcode was installed. Madhavapeddy et al. used SpotCodes to enhance human-computer interaction by using a camera-phone as a pointing and selection device [26]. Researchers working on the CoolTown project at HP Labs proposed tagging electronics around the house with barcodes to be read by camera phones or PDAs so that additional data about the tagged device could be easily retrieved [1].

Hanna considers devices with barcodes affixed to aid in the establishment of security parameters [17]. This work considers a smart home, where a user may want to establish a security context for controlling appliances or other devices in a smart-home. We refer to the security property discussed in this work as *presence*, where it is desirable that only users or devices close to some device are able to control it. We discuss the notion of *presence* further in Section 7.

3 Seeing-Is-Believing

Researchers have worked to find a convenient technology that can facilitate authentication between mobile devices, e.g., [4, 12, 35]. To date, however, proposed systems are either too cumbersome, too confusing, or require excessive diligence and knowledge from users. With SiB, a mobile phone's integrated camera serves as a secure side channel to strongly authenticate one or both parties, thereby bootstrapping secure communications, providing demonstrative identification of the party from which data originates, and capturing user intentions in an intuitive way. What better way for a user to tell device A that it should communicate securely with device B than to take a picture of device B using device A's integrated camera?

3.1 The Visual Channel

We realize the visual channel with a two-dimensional barcode (e.g., Data Matrix [9], PDF417 [37], or MaxiCode [34]), displayed on, or affixed to, one device and captured by another with its digital camera. Section 8 contains details on the barcode implementation employed in SiB. The primary strength of the visual channel is its use as a side-channel for human-verifiable authentication. When a user executes the SiB protocol, he must aim the camera of his mobile device at a barcode on another device (either displayed electronically or affixed to the device's housing). The act of aiming the camera at the desired device results in *demonstrative identification* of the targeted device.

We now present a more detailed example of the use of SiB. Alice's phone generates a 2D barcode encoding appropriate public cryptographic material and displays it on its screen, while Bob uses his phone's digital camera to take a snapshot of Alice's screen displaying the barcode. Bob's intention to communicate with Alice's *particular* device is naturally captured, in that Bob just took a picture of the device with which he wants to establish a secure communication channel. Bob must watch his phone's LCD, acting as viewfinder, updating in real time in response to his positioning of his camera-phone. A machine vision algorithm processes each image in the viewfinder in real time and overlays a colored rectangle around successfully recognized barcodes. When Bob presses the shutter button, the viewfinding process stops and the machine vision algorithm returns the data represented by the barcode. If the image on Bob's LCD is anything other than what he expects, he can abort the protocol. Section 8 presents further details of our implementation.

Active attacks are extremely difficult to perform against SiB without being detected by the user. The user has in mind the device at which he is aiming his camera, and will be conscious of a mistake if he takes a snapshot of anything else. We believe the act of taking a picture of *that* device—the one with which the user wants to communicate securely—is intuitive, and should therefore enjoy a low rate of operator error. Thus, the visual channel has the property that it is secure against active attacks (such as a man-in-the-middle attack), or the property that active attacks are easily detected by the user, who can then terminate wireless communication. It is ideal for use as a location-limited channel for authentication.

3.2 Pre-Authentication

We build on work by Balfanz et al. [4], and Stajano and Anderson [35], to secure wireless communication by leveraging the location-limited visual channel for authentication. We adopt the term pre-authentication, as Balfanz et al. suggest [4], to describe the data exchanged on the location-limited channel, which is later used to authenticate one or both of the communicating parties in almost any standard public-key communications protocol over the wireless link. Eavesdropping on the location-limited channel should give no advantage to an attacker, provided that the underlying cryptographic primitives are secure.

Figure 1 shows the pre-authentication phase of SiB, carried out over the visual channel. Provided that the visual channel and relevant cryptographic primitives are secure against active adversaries (Section 9 presents a detailed security analysis), authentication in our system requires merely that the user confirm his camera is pointed at the intended device.

3.3 Device Configurations

The concepts of SiB can be applied in different ways to devices with different capabilities, each equipped with either a camera and display, a camera only, a display only, or neither. In some cases, these device configurations impose some limitations on the strength of the achievable security properties. Figure 2 contains a summary of these properties.

The most flexible configuration for SiB is when both devices have both a camera and a display—these have a CD in their column or row heading in Figure 2. These devices can be mutually authenticated, since both possess cameras. Further, each device can make use of either a long-term public key or a freshly generated public key in each exchange, since barcodes containing keys are displayed on an electronic screen (as opposed to paper or some other fixed medium).

We discuss devices equipped with no display—"displayless" devices—which have no D in their column or row heading in Figure 2. These devices can be authenticated with a long-term

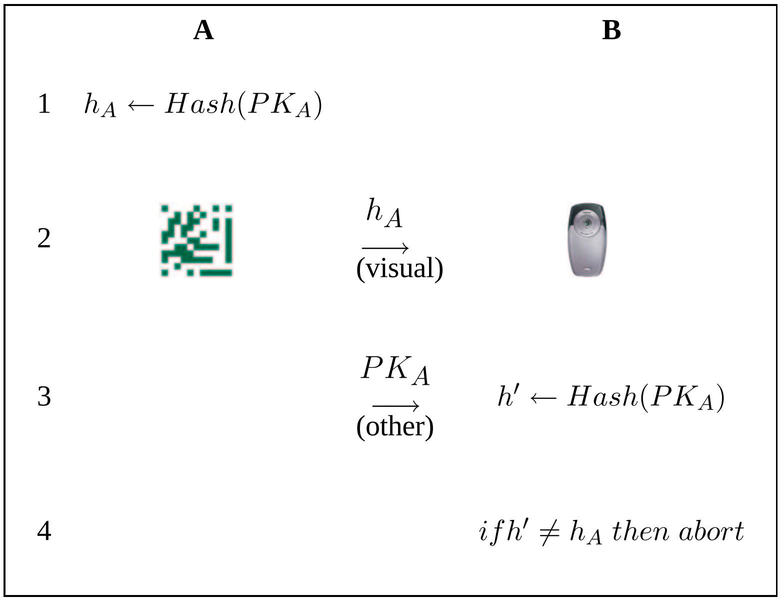

A
B
1
$h_A \leftarrow Hash(PK_A)$
2
$\xrightarrow{h_A}$
(visual)
3
$\xrightarrow{PK_A}$
(other)
$h' \leftarrow Hash(PK_A)$
4
$if h' \neq h_A \; then \; abort$

Figure 1: Pre-authentication over the location-limited visual channel. PK_X is X's public key, which can be either long-term or ephemeral, depending on which standard protocol is selected.

		Y			
		CD	**C**	**D**	**N**
X	**CD**	✓	✓*	✓	✓*
	C	✓	✓*	✓	✓*
	D	presence	presence	×	×
	N	×	×	×	×

Legend	
✓	Strong authentication possible
✓*	Barcode required on housing label
presence	Confirm presence only
×	No authentication possible

Figure 2: Can a device of type X authenticate a device of type Y? We consider devices with cameras and displays (CD), cameras only (C), displays only (D), and neither (N).

public key. A barcode encoding a commitment to the key, or multiple barcodes encoding the key itself, must be affixed to the device's housing (e.g., in the form of a sticker). The issue of whether to use a commitment to a key, or the key itself, is addressed in Section 5.

Entries in Figure 2 marked *presence* indicate that cryptographically secure properties are unattainable, but a property we term *presence* is still achievable. Presences refers to the ability to demonstrate that a device is in view of someone. We describe this property in more detail in Section 7.

4 Bidirectional Authentication Using Seeing-Is-Believing

In this section, we show how SiB can be used to intuitively capture user intentions and establish a mutual security context (e.g., authenticate the exchange of public key certificates, or perform an authenticated Diffie-Hellman key exchange to establish a shared secret) between precisely the devices the user wants, without a trusted authority. One practical example of this key material is a self-signed public key certificate extended with additional information about the key owner (e.g., name, email address, etc.—similar to a vCard [10, 19]). The device combinations we consider in this section are those indicated by a ✓ in Figure 2, since these allow bidirectional authentication.

The SiB protocol begins when Alice and Bob decide they want to communicate securely. Alice's device[2] first discovers nearby devices using its wireless interface. Alice then selects the device she believes to be Bob's (this is easily achieved with "friendly names" common to, e.g., Bluetooth [16]). Since all devices are running a SiB daemon, Bob's device is listening when Alice makes a connection attempt. As the processing and display capabilities of mobile phones improve, visual channel bandwidth will improve sufficiently for an alternative structure to become feasible. Data transmitted over the visual channel could then include network addresses for the relevant wireless interfaces (e.g., Bluetooth, 802.11) in addition to authentication data. This is more convenient for the user, since he or she never has to wait for discovery of neighboring devices or select a device from a list. Madhavapeddy et al. use barcodes on camera phones to improve the Bluetooth device discovery process [27].

Each user computes the digest of his public key material. This key material can take the form of a user's long-term public key certificate, or it can be a freshly generated key for use in only one key exchange for privacy reasons. The decision regarding what form of public key material to use is orthogonal to the authentication provided by SiB.

The pre-authentication phase now begins. The users take turns displaying and taking snapshots of their respective barcodes. The order is not important, but it is necessary that Alice capture the digest of Bob's public key, and that Bob capture the digest of Alice's public key. These digest values serve as commitments from Alice and Bob to their respective public keys. This pre-authentication protocol is secure as long as an attacker cannot find a second preimage for the digest function, and is unable to perform an active attack on the visual channel.

After pre-authentication is complete, both devices now hold commitments to the other device's public key, and devices can exchange public keys over the wireless link. The devices then ensure that the digest function taken over the other device's public key matches the commitment that was received over the visual channel. These authenticated public keys can then be used appropriately in any well-known public key protocol on the wireless link (e.g., IKE [18], SSL [14]). It is imperative that the chosen established protocol verifies that each party does in fact hold the private key corresponding to their authenticated public keys.

If a user desires to avoid transmitting a public key on the wireless network at all—so that eavesdroppers cannot ascertain which devices are communicating—the public key can be encoded in a sequence of barcodes. The key is thereby obtained without transmitting it on the wireless medium, while retaining the demonstrative identification property with respect to the device originating the key. Still, to minimize eavesdroppers' chances of success, the standard public key protocol that is used with SiB authentication must be key-private [5].

[2] Without loss of generality, Bob could just as easily have initiated the exchange. Further, two people need not be involved; a single person may wish to establish a security context between two devices in his or her possession.

5 Unidirectional Authentication Using Seeing-Is-Believing

We now discuss entries from Figure 2 where the device of type X (the authenticator) is equipped with a camera, and the device of type Y (the device being authenticated) lacks a display. It is this presence of a camera, and lack of a display, that are responsible for the security properties of this particular device combination. Thus, we refer to a device of type X as camera-equipped, and a device of type Y as displayless.

Displayless devices do not have the ability to display newly generated values. Still, a camera-equipped device can authenticate displayless devices and establish secure communication channels. The displayless device must be equipped with a public/private key pair, and a sticker containing a barcode of the digest of its public key must be affixed to its housing. Since the displayless device is constrained to the use of a single public/private key pair for its entire lifetime,[3] the option to generate per-interaction public keys no longer applies.

5.1 Practical Applications

An 802.11 access point (AP) is one example of a class of device where “sticker” authentication may be desirable. Camera-enabled devices can authenticate the AP, enabling the establishment of a secure link-level connection from the camera-enabled device to the AP. This solution may be desirable to enable deployment of wireless connectivity in environments where security policies are based on the existence of wired networks. Figure 3 shows the SiB application on a mobile phone scanning a barcode installed on a wireless access point.

Another commonly considered application where demonstrative device identification is desirable is a printer in a public place. Similar to the access point, the printer can have a barcode affixed to its chassis so that a user can use SiB to authenticate the printer and bootstrap establishment of a secure connection.

As mentioned above, one disadvantage of having a fixed public key is that, when the barcode contains a commitment to the public key, the displayless device’s public key gets broadcast in the clear every time the displayless device is authenticated. This enables an attacker to gather data on the usage patterns for the displayless device, which may be undesirable. Thus, if the user of the displayless device is concerned with its usage patterns being tracked, it can never send its public key in the clear.

The solution to this problem is to use a sequence of barcodes on the displayless device which together contain its entire public key, and not just the public key’s digest. Camera-enabled devices can then get the displayless device’s public key without transmitting it in the clear over the wireless link. We discuss multiple-barcode encodings further in Section 9. Note that the public key protocol that is used with SiB authentication must also be key-private [5].

5.2 Establishing a Secure Channel to a TPM

The Trusted Computing Group (TCG) specifies a Trusted Platform Module (TPM) that can be used to enhance the security of many computing platforms [2, 3]. The TPM is a chip connected to a

[3] Of course, devices can be reprogrammed and new stickers affixed, but we consider that to be significant maintenance task.

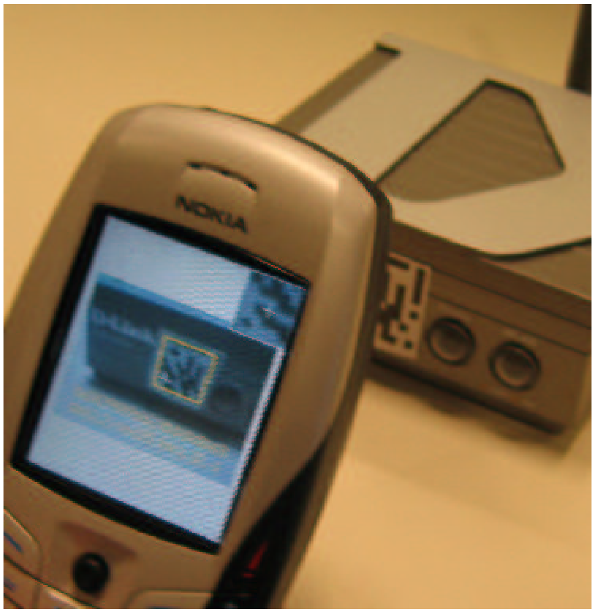


Figure 3: Phone running SiB scanning a barcode on an 802.11 wireless base station.

computer's processor, with no other I/O capabilities. A full discussion of the TCG specification is beyond the scope of this paper, though we summarize relevant concepts here as necessary.

One challenge in designing systems which incorporate a TPM is how a user can communicate securely with the TPM, since the user only has a keyboard and display to communicate with the TPM, with untrusted operating system software in between. A TPM gets configured—typically by a user or vendor—with a secret, the Owner Authorization Data (OAD), which can be used to exercise control over the TPM. A malicious party that captures the OAD can change the OAD, delete (and in some instances change) application secrets in secure storage, and disable or enable TPM features at undesirable times. Unfortunately, in a TPM-equipped desktop computer, if certain TPM features are disabled, a user has no way of knowing if malicious software is running and, e.g., an attacker could be logging all keystrokes. This is a serious problem if the user types the OAD while trying to configure the TPM—the malicious software has just captured the OAD.

Thus, it is undesirable to use the keyboard and display of a computing platform to configure the TPM for fear of malicious software running on the platform that can steal valuable secrets. In the remainder of this section, we show how authentication achievable with SiB enables the user to use his camera phone to send commands to the TPM, achieving secrecy so that a malicious application is unable to capture the OAD even if it has subverted the keyboard and display.

The TCG specifies that each TPM come equipped with a public / private *endorsement* keypair. The public endorsement key is used for encrypting commands to the TPM, and the private endorsement key is used exclusively for their decryption (it is never used to e.g., compute a digital signature). Thus, we propose the use of a camera phone to securely configure the TPM. Using SiB, the camera phone can authenticate a TPM's public endorsement key, and bootstrap secure communication with the TPM through which the user can reconfigure the TPM as desired.

To enable TPM reconfiguration from a camera phone, a sticker must be installed on the case of the platform which contains either a barcode encoding a digest of the public endorsement key, or several barcodes encoding the entire public endorsement key. The performance implications of encoding the entire key into a series of barcodes are considered in Section 8.2. In the case of a digest, the full endorsement key can be obtained from the platform in an authenticated way analogous to the wireless access point example earlier in this section.

Note that an attacker with direct access to the platform can subvert the TPM by physical means. Thus, the use of SiB enhances security under the assumption of software-only attacks, which represent the majority of threats.

6 Demonstration of Screen Ownership Using Seeing-Is-Believing

In this section we consider the problem where a user wants assurance that what he sees on the screen of a TPM-equipped computer is really the content displayed by a particular application—we want to determine that this application *owns* the screen. We propose to use SiB in conjunction with the TPM to assist in determining whether the content displayed on a computer's screen truly originates from a particular application. Our solution prevents a malicious application from impersonating a legitimate application on the local machine. For example, a user's machine may have a virus that tries to spoof a legitimate control panel application to get the user to enter their password. Note that, if the local application is a web browser, our techniques put in place a framework that can be highly effective against web spoofing [13, 43].

The TCG specifies that the TPM has Platform Configuration Registers (PCRs) that a properly instrumented operating system extends[4] with hash values computed over all software that is loaded for execution. This includes the boot process, which the TCG terms "trusted boot." The purpose of the PCRs is to enable the TPM to digitally sign attestations that a particular software image has been loaded. We extend this concept to screen ownership below.

We include a brief note on terminology. The TPM is a passive component, so application-level commands do not actually get sent to the TPM—they are interpreted by a piece of trusted software that can be attested to, usually a module in the operating system. This trusted software translates the application-level commands into TPM hardware instructions. Subsequently, when we say the user sends a command "to the TPM," we mean that it is sent to this special piece of software, and, if necessary, the TPM can be used in an attestation that the OS module has not been modified.

6.1 Protocol Description

Several prerequisites must be met to verify that the desired application actually owns the screen. The operating system and the windowing system (i.e., the window manager) must be designed such that they can enforce the condition that the desired application's window is "always-on-top." There are additional details associated with the demarcation of window boundaries, e.g., that the "always-on-top" window must have borders of a substantially different color than anything else on the screen; however, these details are beyond the scope of this paper. For the remainder of this section, we assume the existence of an appropriate windowing system.

We achieve through demonstration of screen ownership the property that the consistency of a particular application can be verified over time. For example, we can verify that the current

[4] For details on the "extension" process, please refer to the TCG Specification [2].

application is, e.g., the same application we were using last week. This is different from the property of confirming that the current application is, e.g., precisely the application purchased from a particular software vendor. The techniques described below can easily be extended to achieve such a property, however, additional vendor-specific configuration is necessary on the user's camera phone which is outside the scope of SiB.

The user's application need not be certified. The spirit of operation is similar to that of a common SSH session—upon first connection keys are exchanged under the assumption that an attack is highly unlikely. If this assumption holds, then all subsequent connections are secure.

We now describe the basic operation of SiB on a TPM-equipped machine in two parts—the initial configuration, and subsequent verifications. We omit many details for brevity.

6.1.1 Initial Configuration

We explain the initial configuration under the assumption that there are no attacks in progress. We assume the user has already established a secure channel to the TPM via Bluetooth, as in Section 5.2. The user then elects to do an initial configuration of the active application on the TPM-equipped computer by selecting the appropriate menu item on his mobile phone. Upon reception of the command to do an initial configuration of the active application in the TPM-controlling piece of the OS, the following occurs:

1. The window manager temporarily locks out other applications, and ensures that the active application remains "always on top" for the duration of the protocol.

2. The TPM allocates a new secure data object for storage of the current values of the PCR registers for the OS, window manager, and active application.

3. The TPM allocates a new secure key object and generates a new public / private key-pair K_{verify} to be used in subsequent verifications of the active application.

4. The TPM sends the public K_{verify} to the user's mobile phone, along with appropriate meta-data such that the user can easily select the correct application for performing subsequent verifications.

6.1.2 Subsequent Verification

We leverage the ability of the TPM to conditionally release a private key [3]. The condition is that the appropriate PCR registers contain the same values that they did upon initial configuration. That is, the same operating system, window manager, and application are running now that were running during initial configuration.

The user elects to do a verification by selecting verification of the appropriate entry from a list of applications on his mobile phone (an entry is added to this list during each initial configuration, we omit the details for brevity). Note that the user need not establish a secure channel to the TPM via Bluetooth, in contrast to the initial configuration. Malicious software executing on the platform being verified may be able to capture the challenge, but it cannot access the TPM-protected private key K_{verify}^{-1} necessary to compute a valid signature. The verification then proceeds as follows:

1. The phone generates a cryptographically secure nonce and sends it over the secure channel to the TPM, indicating which application it should be delivered to.

2. The TPM then checks whether the appropriate PCR registers for the specified application, window manager, and OS contain the values recorded during the initial configuration.

3. If the correct values are present, the private verification key K_{verify}^{-1} is released to the specified application. The challenge received from the user's mobile phone is also delivered to the specified application.

4. The application is then able to sign the challenge using the private verification key: $\sigma \leftarrow Sign_{K_{verify}^{-1}}(Challenge)$.

5. The application encodes σ in one or more barcodes and displays those on the screen.

6. The user uses the SiB component of the application on his camera phone to capture this signature and verify it with K_{verify} as obtained in the initial configuration: $Verify_{K_{verify}}(\sigma')$.

7. If the signature verifies, the user has a cryptographically strong guarantee that the active application on his computing platform is precisely the same application that was active when he performed the initial configuration.

6.2 Extension to Operate without Bluetooth

If either of Bob's mobile phone, or Bob's computer, is not equipped with a Bluetooth interface, we can still achieve strong security properties. This comes at the expense of convenience for Bob. The security-critical role of Bluetooth in the protocol was to deliver the cryptographic challenge to the TPM. Without Bluetooth, Bob must do this. Basic operation proceeds as follows:

1. Bob presses a button on his phone indicating he wishes to do a verification. The phone then generates the nonce that will be used in the challenge and displays it, or some user-friendly encoding of it, on the phone.

2. Bob selects a menu option on his computer so that it enters the verification protocol (previously this happened automatically upon reception of the appropriate message via Bluetooth). Bob then types the challenge displayed on his phone into the application on his computer that currently owns the screen. Assuming no attacks, the PCR registers contain the appropriate values so the TPM releases the signing key K_{verify}^{-1} to the application. The application can then sign the challenge and display the signature as a barcode, which Bob can then verify with SiB as before. If a malicious application owns the screen, the TPM will refuse to release the appropriate K_{verify}^{-1}, and the malicious application will be unable to construct a signature that will pass verification on Bob's phone.

7 “Presence” Confirmation

A display-only device (display-equipped and cameraless) is unable to strongly authenticate other devices. Equipped with no camera, it makes no difference whether the entity the cameraless device wants to authenticate has a display, or makes use of a barcode sticker—the cameraless device cannot see them. However, cameraless devices can obtain a property we refer to as “presence” (see Figure 2). That is, it can confirm the presence of some other device in line-of-sight with the display.

To detect the presence of a nearby device, the display-only device generates a nonce value, encodes it in a barcode, and displays that barcode, noting the time when it was first displayed. Any nearby devices that are able to see the display and capture the barcode should send the nonce value back to the display-only device over the wireless channel. When the nonce arrives over the wireless channel, the device knows the somebody has been in line-of-sight during the time since the nonce was first displayed. We emphasize that this “presence” property is quite weak—the display-only device has no way of knowing how many devices can see its display, or whether the radio signal from the device in line-of-sight with the display was tampered with. It can only verify that the value it receives over the wireless channel is the same as the value it displayed as a barcode, and it can measure the length of time between displaying the barcode and receiving the nonce on the wireless channel.

Despite the weakness of the “presence” property, there are still practical applications for devices capable of determining “presence.” For instance, the “presence” property is useful in the context of a smart home. It can restrict access to channel control on a television to users in the same room. In general, it can serve to limit authority to control a device to users located in view of that device. This is similar to the scenario considered by Hanna [17]. Hanna considers devices with barcodes affixed to aid in the establishment of security parameters, where a user may want to establish a security context between appliances in the home and a central home security controller. In Hanna’s work, the barcode contains a secret which is also stored inside the device. Hanna proposes using this secret to enable the secure transmission of commands to the device over an untrusted network.

8 Implementation Details

8.1 Series 60 Phone Application

We built SiB in C++ such that it will run on mobile phones running Symbian™ OS versions 6.1 and 7.0s with the Nokia™ Series 60 User Interface. The size of the Symbian Installation System (SIS) file [38] for SiB is only 35 kilobytes. This makes deployment feasible over even the most constrained channels, such as GPRS [33]. We performed 10 trial downloads of the SiB application over GPRS using an AT&T™ mMode™ account and a Nokia™ 6600, achieving a maximum download time of 5 seconds, a minimum of 4 seconds, and an average of 4.5 seconds.

The Nokia™ 6600 served as our development platform. The barcode format and image processing algorithm in our system is adapted from that of Rohs and Gfeller [32]. The data contained in the barcodes for SiB is augmented with Reed-Solomon error correcting codes to provide better performance in the presence of errors in the image processing [31]. We ported Karn's implementation of Reed-Solomon codes to Symbian OS [21].

In its current state, we use the SHA-1 cryptographic hash function for all hashing operations. All wireless communication occurs via Bluetooth [16]. When the user elects to perform a key exchange between two camera phones, we implemented ephemeral Diffie-Hellman key exchange to establish a shared secret between the two devices [11, 20]. This shared secret can be used to establish an authenticated channel over the wireless data link between the two devices, over which any desired information can be exchanged.

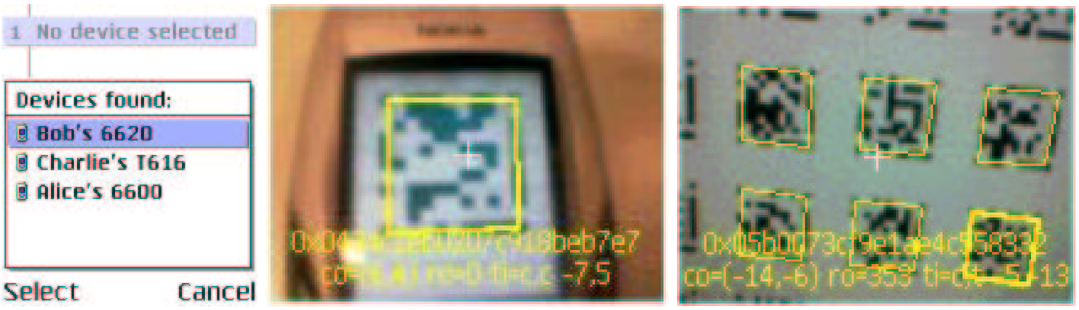
Since the Reed-Solomon codes embedded in the barcode indicate whether a processed code was valid or invalid, and our application constantly decodes any barcodes in the current camera scene, it is not strictly necessary for the user to press a "shutter" button for the camera. Our implementation is configurable so that the user may elect whether he wishes to press a shutter button. When the shutter button is disabled, the first valid image processed can be used automatically. However, if the user is in an environment where there are many barcodes, recognition of the incorrect barcode will cause the SiB protocol to abort. For particularly cautious users, who may be concerned that automatic shutter control could cause the camera to capture a malicious barcode before the user aims the camera at the desired barcode, manual control of the shutter is essential.

Figure 5 contains photographs of SiB in action. Figure 5(a) shows Alice running the SiB application to perform a key exchange—her phone is displaying a barcode on a Nokia™ 6600. Figure 5(b) shows Bob's Nokia™ 6600 running the SiB application, successfully decoding the data encoded in Alice's phone's barcode. In bidirectional authentication with SiB, Alice and Bob would then switch roles. Bob's phone would display a barcode, and Alice's phone would decode it. Figure 4 contains screen shots of the SiB application. In Figure 4(a), the user is being prompted to select the desired Bluetooth device for the key exchange. We emphasize that device selection is for convenient establishment of the wireless channel only, and is not a factor in the security of SiB. Madhavapeddy et al. detail the encoding of Bluetooth network addresses to barcodes displayed on mobile phones to eliminate the Bluetooth discovery process [27]. Implementation of SiB using a barcode with sufficient bandwidth to encode a Bluetooth network address and a cryptographically secure hash value can eliminate the manual device selection process. Figure 4(b) shows the display of one phone when it has successfully recognized the barcode displayed on the screen of another phone.

8.2 Barcode Reading Performance

In this section we evaluate the performance of our SiB application on the Nokia 6600. We consider the length of time required for SiB to mutually authenticate two mobile phones and perform a Diffie-Hellman key exchange, as in Section 4. We then provide insight into the practicality of using multiple barcodes to encode a single logical item. For example, a 1024 bit RSA key would need to be encoded in 16 barcodes in our current implementation—64 bits of key and 4 bits of place-marker in each barcode.

We performed a small user study with our implementation of bidirectional authenticated Diffie-Hellman key exchange application (see Section 4) between two Nokia 6600s. We instrumented the application to track the length of time between the establishment of the Bluetooth connection and the successful completion of the Diffie Hellman key exchange. It is reasonable to

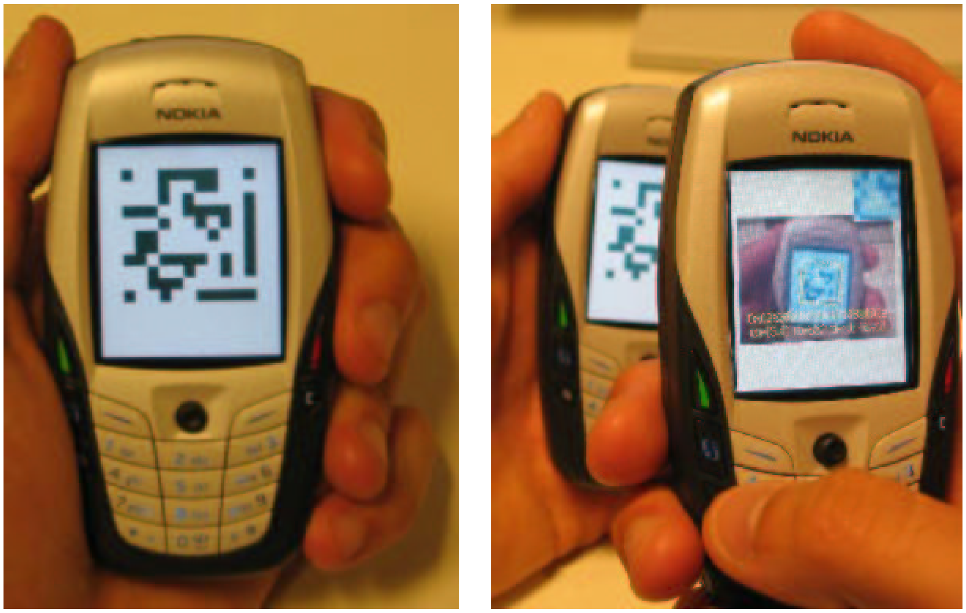


(a) Device selection.

(b) Barcode recognition.

(c) Multiple barcode recognition.

Figure 4: Mobile phone screen shots showing the SiB application in operation. In the first screen shot, the user is being prompted to select a device with which to initiate the SiB protocol. In the second, the user sees the phone at which he is aiming his camera, with status markers indicating successful barcode recognition. The third image is successful recognition of multiple barcodes displayed on an LCD screen.



(a) SiB application displaying a hash encoded in a barcode.

(b) SiB application with one phone scanning the hash-barcode on the LCD of another.

Figure 5: SiB protocol in action.

ignore Bluetooth device discovery since a commercially viable implementation should use a barcode format with sufficient bandwidth to include the Bluetooth address of the displaying device, rendering device discovery unnecessary. Over a series of 10 executions involving manual aiming of the phones, we observed the average length of time to establish a shared secret to be 8 seconds. The minimum and maximum times ranged between 6 and 10 seconds, respectively.

On the Nokia 6600, SiB is able to process two to three barcode snapshots per second. We have successfully read in excess of five barcodes from a single snapshot, for a sustainable rate averaging 10 to 15 barcodes per second under ordinary office lighting conditions. Thus, we conclude that reading multiple barcodes for a single logical item is a viable implementation strategy.

9 Security Analysis

In addition to the security of the underlying cryptographic primitives, the security of SiB is based on the assumption that an attacker is unable to perform an active attack on the visual channel. We first discuss the employed cryptographic primitives; then the security properties of various side-channels for authentication. Finally, we discuss the security properties of authentication using a visual channel.

9.1 Cryptography

As implemented, each barcode in SiB has a raw bandwidth of 83 bits. 15 of these bits are dedicated to Reed-Solomon error correcting codes, leaving the application with 68 bits of hash for authentication. As discussed in Section 4, the hash transmitted in the barcode needs to be secure against active attacks, which we achieve through the properties of the visual channel. However, if an adversary can find a second pre-image of the value encoded in the barcode, then a passive attack on the barcode coupled with an active attack against the wireless network connection can be successful.

For a cryptographic system to be considered computationally secure today, at least 2^{80} operations should be required to break it [23]. When the barcode containing a key's digest takes the form of a sticker on the side of some device, or gets displayed on a laptop or other relatively large screen, it is not a problem to use multiple barcodes to encode sufficient cryptographic material to be secure based on standard cryptographic assumptions. As detailed in Section 8.2, SiB is capable of reading multiple barcodes simultaneously.

However, in phone-to-phone key establishment, there is only enough screen area to display a single barcode. Thus, the 68 bits of hash data in our key exchange prototype are potentially vulnerable to cryptanalytic or brute-force attack. However, this vulnerability is significant only when the barcode represents a hash of a long-term public key. We propose two methods for circumventing this problem. The first simply involves the use of multiple barcodes to encode sufficiently many bits of hash, displayed in sequence. The phones can synchronize the displaying and reading via Bluetooth messages so that the user is not inconvenienced by pressing buttons to iterate through multiple barcodes. While effective, this technique adds to the burden the user must face.

We propose a second technique which still achieves strong security properties through the use of Diffie-Hellman session keys. The devices participate in an ephemeral Diffie-Hellman key exchange, using SiB to authenticate their respective contributions to the key. This session key is then used to establish an authenticated channel over which the devices may exchange all manner

		Resists	
Channel	**COTS**	**MITM**	**Convenient**
Ultrasound	○	○	●
Audible (“beeps”)	○	◐	●
Radio	●	○	●
Physical Contact	○	●	●
Wired Link	●	●	○
Speaking Passwords Aloud	●	●	○
Writing Passwords Down	●	●	○
Visual Hash Verification	●	●	◐
Infrared	●	◐	◐
Machine Visual (SiB)	●	●	●

Legend	
●	Yes
◐	Partial
○	No

Figure 6: Characteristics of various side channels proposed for authentication. We acknowledge that rating the convenience of a channel is subjective; however, we believe it is useful to compare various channels in this way. COTS indicates that the necessary hardware is already present in Commercial Off-The-Shelf products.

of data. Once the DH session-key establishment is complete, there is no advantage to finding a pre-image or second pre-image of the hash function.

Even with today’s technology, computing 2^{67} operations takes well over a few seconds, which is the length of time DH key establishment using SiB for authentication requires to execute.[5] Once a session key is established, the attacker gains nothing even if he compromises the hash. In a commercially viable system, the barcode generator and recognition algorithm should be extended to achieve a useful data content of 80 bits or more. There are no major technical obstacles preventing this extension. We emphasize our focus on the attainable security properties of a complete system, and not the development of image processing algorithms.

9.2 Selecting a Side Channel

Mutual authentication between two parties without the assistance of a trusted authority requires a side-channel that is secure against active attacks, such as a man-in-the-middle attack. We analyze potential side channels based on the degree to which the user’s intentions are captured, and the amount of feedback that the channel provides to the user. Figure 6 contains a summary of proposed side-channels and their characteristics.

Activity on channels such as infrared, ultrasound, or radio is undetectable to humans without specialized equipment. Therefore, if Alice believes her device is communicating with Bob’s device via infrared, the only assurance she has that it is actually doing so is through status indicators on the two devices. She cannot see infrared radiation leaving her device and entering Bob’s, and she certainly cannot see Mallory’s device outputting interference patterns and affecting the data stream. Similarly, in case of ultrasound and radio, Alice and Bob need to rely on status indicators

[5] The processing time required by SiB is under one second. The execution time of SiB is dominated by the users aiming their cameras at the correct barcodes.

of their devices, but they are not sure that Alice's device is indeed setting up a key with Bob's device. Thus, the users' intentions are not captured well, and feedback is indirect and prone to error. Using an audible signal (marked "beeps" in Table 6) for data exchange is more intuitive, but this would not work well in noisy environments and is still prone to a man-in-the-middle attack since it is hard for people to tell where "beeps" originate and how many devices are "beeping."

Physical contact between devices is much more intuitive for people and captures the intention of the users with which device they want to establish a secure communication link [35]. Unfortunately, current devices are not equipped with an interface for this purpose. An alternative approach is to use a wired link, for example connect both devices with a USB cable, however, this approach is not convenient to use and people would need to carry a wire with them. In addition, connecting the two devices may introduce a security vulnerability, for example, consider a malicious device mimicking a USB plug-and-play device that installs software on the target device.

Another approach is for Alice and Bob to establish a secret password, either by speaking the password aloud, or by writing passwords on paper and passing them to each other. Both Alice and Bob would then need to type in the password correctly, which the devices use to perform a secure password protocol, e.g., EKE [6]. We believe this approach is cumbersome in comparison with SiB, and in addition, would not work for devices without a keyboard.

Finally, both devices could present a visual representation of the hash of the established Diffie-Hellman key to detect a man-in-the-middle attack [12, 15, 24, 30]. These approaches, however, are not secure unless people carefully compare the output of the visual hash function. We believe SiB has an advantage here not just in ease-of-use but because strong authentication is intrinsically linked with device identification.

9.3 The Visual Channel (Camera and Display)

A more secure channel for purposes of authentication is one that captures the user's intentions. The visual channel provides several desirable security properties. The first is that active attacks are extremely difficult to perform. Second, taking pictures is a familiar and intuitive action.

9.3.1 Resistance to Active Attacks

An attacker can disrupt the lighting conditions around Alice and Bob in an attempt to disrupt SiB. However, changes of sufficient magnitude to impair SiB are easily observed by Alice, Bob, and any people in the vicinity, alerting them to some kind of unusual behavior. A more sophisticated, and subtle, attack is to use infrared radiation to overwhelm the CCD[6] in a phone's camera. If an attacker is able to flood an environment with sufficient infrared radiation, the CCD in a phone's camera can begin to saturate, and all attempts to take pictures will yield a picture with all pixels set at or above the intensity of the legitimate image, up to the maximum value for each pixel. Essentially, the image becomes noise. Alice will see that the image in her viewfinder is not the picture of Bob's phone that she expects, and can abort the protocol.

Even without a user monitoring the process, the electronic-warfare-esque techniques necessary to cause the CCD to output something other than the scene in front of the camera are beyond the

[6] Charge Coupled Devices (CCDs) are the prevalent type of image sensor used in today's digital cameras.

reach of all but the most sophisticated adversaries with current technology. We are unaware of any attacks feasible today which result in anything but noise from the camera under attack.

9.3.2 Usability

The primary strength from a security perspective that is gained by using cameras and barcodes is that the user can visually identify the other communication endpoint. Alice can see the barcode displayed on the screen of Bob's phone, and can see the same barcode on her own screen while it functions as a view-finder for the camera. This interface is intention-driven, and it provides demonstrative identification of the device whose picture is being taken. Any attempts to tamper with the image received by Alice's phone will be immediately apparent to Alice.

SiB achieves a high degree of security while requiring less diligence from users than existing schemes. Rather than comparing details of a graphic (e.g., [12, 30]), the user is only concerned with aiming the camera at the appropriate device.

10 Conclusion

In this paper we propose Seeing-Is-Believing, a system that uses machine vision as a location-limited side-channel for human-verifiable authentication. This side-channel rules out man-in-the-middle attacks against public-key based key establishment protocols. It also enables a TPM-equipped computer to demonstrate control of the computer's display, preventing software-based attacks. The visual channel has the desirable property that it provides demonstrative identification of the communicating parties, providing the user assurance that his or her device is communicating with *that* other device. We have also analyzed the establishment of secure, authenticated sessions between SiB-enabled devices and devices missing either a camera, a display, or both, and found that secure communication is possible in many situations.

References

[1] CoolTown. `http://www.cooltown.com/`, November 2004.

[2] Trusted computing group. `http://www.trustedcomputinggroup.org`, November 2004.

[3] Boris Balacheff, Liqun Chen, Siani Pearson, David Plaquin, and Graeme Proudler. *Trusted Computing Platforms – TCPA Technology in Context*. Prentice Hall, 2003.

[4] 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)*, San Diego, California, February 2002.

[5] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Advances in Cryptology—Asiacrypt*, volume 2248 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001.

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

[7] 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.

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

[9] RVSI Acuity CiMatrix. Data matrix barcodes. Available at: `http://www.rvsi.com/acuitycimatrix/index.htm`, last verified August, 2004.

[10] F. Dawson and T. Howes. vCard MIME directory profile. RFC 2426, September 1998.

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

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

[13] Edward W. Felten, Dirk Balfanz, Drew Dean, and Dan S. Wallach. Web spoofing: An internet con game. Technical Report 540, Princeton University, February 1997.

[14] Alan O. Freier, Philip Karlton, and Paul C. Kocher. The SSL protocol version 3.0. Available at: `http://wp.netscape.com/eng/ssl3/draft302.txt`, November 1996.

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

[16] J. C. Haartsen. The bluetooth radio system. *IEEE Personal Communications Magazine*, pages 28–36, 2000.

[17] Stephen R. Hanna. Configuring security parameters in small devices. Internet draft, Sun Microsystems, Inc., July 2002. draft-hanna-zeroconf-seccfg-00.txt.

[18] D. Harkins and D. Carrel. The internet key exchange (IKE). RFC 2409, November 1998.

[19] T. Howes and M. Smith. MIME content-type for directory information. RFC 2425, September 1998.

[20] P. Jones. US secure hash algorithm 1 (SHA1). `http://www.faqs.org/rfcs/rfc3174.html`, September 2001.

[21] Phil Karn. Reed-solomon encoding/decoding. Available at http://www.ka9q.net/code/fec/, 2002.

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

[23] 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.

[24] Raph Levien. PGP snowflake. Personal communication, 1996.

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

[26] Anil Madhavapeddy, David Scott, Richard Sharp, and Eben Upton. Using camera-phones to enhance human-computer interaction. In *Sixth International Conference on Ubiquitous Computing (Adjunct Proceedings: Demos)*, 2004.

[27] Anil Madhavapeddy, David Scott, Richard Sharp, and Eben Upton. Using visual tags to bypass bluetooth device discovery. In *ACM Mobile Computing and Communications Revire (MC2R)*, January 2005.

[28] 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.

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

[30] 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.

[31] Irving S. Reed and Gustave Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 1960.

[32] Michael Rohs and Beat Gfeller. Using camera-equipped mobile phones for interacting with real-world objects. *Advances in Pervasive Computing*, pages 265–271, April 2004.

[33] Peter Rysavy. General packet radio service (GPRS). *GSM Data Today*, September 1998.

[34] United Parcel Service. MaxiCode. `http://www.maxicode.com/`, last verified August, 2004.

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

[36] Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An authentication service for open network systems. Manuscript, March 1988.

[37] Symbol Technologies. PDF417. `http://www.pdf417.com/`, last verified August, 2004.

[38] Alexander Thoukydides. Sis file format. Available at: `http://homepage.ntlworld.com/thouky/software/psifs/sis.html`, 2004.

[39] J. D. Tygar and A. Whitten. WWW electronic commerce and Java Trojan horses. In *Proceedings of the 2nd USENIX Workshop on Electronic Commerce*, pages 243–250, Oakland, California, November 1996. USENIX.

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

[41] Simon Woodside. Read real-world hyperlinks with a camera phone. Available at `http://semacode.org/`, last verified August 24, 2004.

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

[43] E. Ye and S. W. Smith. Trusted paths for browsers. In *11th Usenix Security Symposium*, August 2002.

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