The Untrusted Computer Problem and Camera Based Authentication Using Optical Character Recognition

by

Thomas Kotwal

B.S., Electrical Engineering, Tufts University (2000)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2002

© Massachusetts Institute of Technology 2002. All rights reserved.

Author ........................................
Department of Electrical Engineering and Computer Science
May 10, 2002

Certified by .........................

Srinivas Devadas
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by ....................

Arthur C. Smith
Chairman, Department Committee on Graduate Students
The Untrusted Computer Problem and Camera Based Authentication
Using Optical Character Recognition

by

Thomas Kotwal

Submitted to the Department of Electrical Engineering and Computer Science
on May 10, 2002, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

Abstract

The use of a public computer requires an implied trust in that computer to carry out
the user’s requests correctly. When attempting to perform a secure transaction using the
computer, there should be no trust at all placed in the computer. This thesis presents the
problem of using an untrusted computer, proposes a number of possible solutions to the
problem, and discusses the implementation of a specific solution that entails verification of
the screen content using optical character recognition techniques.

Thesis Supervisor: Srinivas Devadas
Title: Professor of Electrical Engineering and Computer Science
Acknowledgments

I would like to thank my advisor, Srinivas Devadas, for his advice and support throughout the completion of this thesis. Many thanks also go to Blaise Gassend, Dwaine Clarke, Matt Burnside, Maren van Dijk, and Ronald Rivest for their invaluable collaboration and guidance in this project. I would also like to thank Larry Rudolph and Ken Steele for lending me an iPAQ and camera, without which the project could not have been completed.

To everyone around the lounge on the second floor of LCS, thank you for keeping things entertaining with seemingly constant discussions of baseball, or games of wiffle ball. Many, many thanks go to my family and friends, whose support and encouragement kept me going throughout my entire education. I could not have done it without you. Thank you.
## Contents

1 Introduction ...................................................... 13
   1.1 Motivation .................................................. 14
   1.2 Model ....................................................... 14
   1.3 Goals ....................................................... 15
      1.3.1 Authenticated Bidirectional Communication ............. 16
      1.3.2 Functionality .......................................... 16
      1.3.3 Ease-of-Use .......................................... 17
   1.4 Related Work ................................................ 17
      1.4.1 Secret Knowledge ...................................... 17
      1.4.2 Smart Card .......................................... 18
      1.4.3 Transparency ........................................ 18

2 Device ............................................................ 21
   2.1 Relaying Device ............................................. 21
   2.2 Monitoring Device ......................................... 22
      2.2.1 Pixel Mapping Approach ............................... 23
      2.2.2 Proxy-Based Approach ................................ 25
      2.2.3 Evaluation and Comparison of Approaches ............ 27

3 Implementation .................................................. 29
   3.1 Overview .................................................. 29
   3.2 Image Estimation .......................................... 30
      3.2.1 Lens Distortion ..................................... 31
      3.2.2 Border Identification ................................. 33
      3.2.3 Projective Distortion ................................ 35
3.3 Optical Character Verification ............................................. 36
  3.3.1 Requirements ......................................................... 37
  3.3.2 Template Matching .................................................. 38
  3.3.3 Constraints ............................................................ 38
  3.3.4 Segmentation ........................................................... 39
  3.3.5 Templates Used ...................................................... 40
3.4 Results and Evaluation .................................................... 43

4 Conclusion ................................................................. 47
# List of Figures

1-1 The Untrusted Computer Model ........................................ 15
2-1 Pixel Mapping Approach Example .................................... 24
2-2 Proxy Based Approach Example ...................................... 28
3-1 Screen Picture Example .................................................. 31
3-2 Demonstration of Radial Distortion ................................. 32
3-3 Screen Picture Before and After Lens Distortion Correction .... 33
3-4 Border Isolation ............................................................ 35
3-5 Restored Image .............................................................. 37
3-6 Sample Character Window ................................................. 40
3-7 Sample Template ........................................................... 41
3-8 Distinction Between Characters ........................................ 44
List of Tables

3.1 Point Assignment for Template Matching Combinations .................................. 42
Chapter 1

Introduction

When using a publicly accessible computer to perform transactions, there is an implied trust in the computer to carry out the transactions correctly. Further, when the transactions require some level of security, such as transactions of a financial nature, the user is trusting the computer to carry out the transaction both correctly and without deceit. The computer could easily display messages that make the user believe that her desired transaction is being performed, while the computer is actually performing a different transaction.

Consider Alice, who is waiting in an airport terminal for her flight to board. She realizes that she forgot to pay her credit card bill and her payment is due the next day. Noticing a few public computers, she decides to pay her bill online by transferring money from her bank account to the credit card company. She sits down at a computer that happens to be infected with a virus that is able to detect access to bank accounts. The virus is able to transfer all the money from the bank account to a separate account owned by the writer of the virus. The virus is also able to make Alice believe that her transaction is being processed correctly, by displaying messages that Alice would expect to see on the screen. Believing that her credit card bill has been paid, she gets on her plane and only finds out the next day that her account has been emptied. By this time the thief has removed all traces of his software from the computer and has made a clean getaway.

For the user to be able to perform transactions without any implied trust in the public computer, the information that the user receives, namely the screen content, must be authenticated. Also the user must be sure that the information that she provides arrives at its destination without modification. This document discusses a general solution to the
problem, and a specific implementation that uses a small, trusted device that is equipped with a simple camera. This implementation also involves a method of optical character recognition designed specifically for this problem.

1.1 Motivation

The development of the internet has changed and is changing the way in which people are using computers. One major trend in this path is the increasing availability of public computers, such as in internet cafes, airport lounges, or hotel lobbies. As this trend continues we can expect public terminals to be even more prevalent, found anywhere from companies' lobbies, to fast food restaurants, to kiosks at street corners. It is conceivable that a basic public terminal will replace the public telephone. Also on the rise is the amount of internet traffic that requires security. More and more people are purchasing stocks and doing some, or even all, of their banking using their computer. The first "cyberbank", First Virtual, was opened in 1994, and the first full service, internet-only bank, First Internet Bank of Indiana, opened in 1999 [7]. The amount of online banking that is taking place has increased at an alarming rate, to a point where today it is hard to find a bank that does not offer online services. As the internet becomes more pervasive, and the amount of internet traffic requiring some form of security increases, the ability to perform secure transactions from an untrusted public terminal becomes increasingly vital.

The system presented in this document requires that the user possess a small, trusted device. This is a reasonable assumption, given the increasing prevalence of devices such as cell phones and Personal Digital Assistants (PDAs). Currently available cell phones or PDAs would require only minor modifications to emulate the device required to enable trusted transactions on an untrusted computer.

1.2 Model

In order to describe the untrusted computer problem in a consistent manner, a model of the situation is given below. This model is heavily based on the model presented in [3]. Figure 1-1 shows a diagram of this model.

- We will assume that a user is attempting to communicate with a proxy, which is a
Figure 1-1: The Untrusted Computer Model. The arrows indicate lines and direction of potential communication.

- The user is in possession of a device that has the capabilities of a computer. We will attempt to limit these capabilities to minimize size, cost, and power consumption.

- The user and proxy are connected through a channel. The channel consists of an untrusted computer to which the user has access as well as an untrusted network. In general, this channel will simply convey messages between the user and proxy. However, the channel is able to exhibit any behavior that is possible of any combination of computers and humans. Note that the channel cannot break cryptographic primitives.

- The user will send and receive messages over the channel, and sometimes will be able to accept or reject the received messages.

- The combination of the user and device will have a shared secret with the proxy, in order to perform cryptographic operations.

1.3 Goals

For a system to be a viable solution to the challenge of allowing secure use of an untrusted computer there are a number of goals that must be met. Described below are the three major goals: authenticated bidirectional communication, functionality, and ease-of-use.
1.3.1 Authenticated Bidirectional Communication

The most important goal of our system is to allow secure communication between the user and proxy. In this case secure communication means authenticated bidirectional communication. In other words, the user and proxy must be sure that the information that each receives was sent by the other party, and was not modified by the channel. Authentication of this nature is a requirement for the user to be able to perform transactions with the proxy and be sure that they are carried out correctly.

In security applications, privacy of information is a commonly sought-after goal. Privacy means that information that is being communicated is kept secret. In general, privacy is not available when using an untrusted computer, however an exception is explained in Section 2.1. The untrusted computer has access to any information that the user enters directly into the computer, as well as any information that it is displaying to the user. This is unfortunately unavoidable, and must be kept in mind when considering the application and the user. For example, consider Alice, who is now using an untrusted computer to manage her stocks. The untrusted computer will know exactly which stocks Alice reviewed, which ones she purchased or sold, and how much money the stocks were worth. Alice may not be concerned about the disclosure of her stock portfolio, in which case the lack of privacy is not an issue. However, she will surely want her transactions to be carried out correctly, which this paper insures by providing a means of authenticating both Alice and her bank to one another.

1.3.2 Functionality

Given that authenticated bidirectional communication can be established between a user and her proxy, the limits on the functionality of the system must be analyzed. The system must be able to authenticate enough information in a short enough time to be useful to the user for a given application. For example, a system that can authenticate 10 KB per second may be adequate for sending and receiving emails, but not streaming video. Also, the type of content that can be verified is important. It can be assumed that there will be some limit placed on the allowed content, such as the maximum resolution of images that can be verified, or the longest text string that can be verified in a single message. Clearly, a system with few restrictions would be preferred to a system with many.
1.3.3 Ease-of-Use

Establishing authenticated bidirectional communication and enabling sufficient functionality for a given application will dictate if that application is feasible; however, the ease-of-use will govern if it is actually used. Factors affecting ease-of-use include the size of the device that the user is required to carry, and the complexity of the procedure required to establish authentication. For example, if the device is large or heavy, the user will be more apt to leave it at home and avoid using an untrusted computer entirely. Similarly, the system will not be used frequently if the procedure required to establish authentication is complicated and requires a significant amount of effort for the user.

1.4 Related Work

A very prevalent and commonly used example of an untrusted computer is an Automatic Teller Machine (ATM). As described in [2], there have been a number of cases in which an ATM has been used to successfully deceive the user and steal her money. In one case the thieves bought an ATM, programmed it to steal users’ personal identification numbers (PINs), and placed it in a public location. Users would enter their pin, at which point they are recorded for later use by the thieves. Then the ATM would display messages indicating that the users’ requests could not be carried out for some reason. Another group went to the extent of constructing an entire false bank branch, and were able to steal $375,000 from unsuspecting users. The following areas of related work are compared by their ability to prevent ATM fraud, and categorized by the method of solution.

1.4.1 Secret Knowledge

A number of methods have been developed that can be used in an attempt to counteract fraud of this nature by doing away with PINs. In some schemes [8, 9, 10] the user identifies herself by using a challenge-response protocol that can only be successfully completed if the user knows a certain secret. A common complaint of these systems is that the user’s secret can be long and difficult to remember. This complaint is addressed in [5] where identification is provided by requiring that the user recognize a series of previously viewed images. While these methods enable the use of an untrusted ATM by eliminating the need for PINs, they are only able to provide identification of the user. They do not accomplish
the goal of providing authentication of the user, the proxy, and the information that is exchanged between the two.

1.4.2 Smart Card

A system that comes a step closer to our goal is described in [1], in which a smart card is used to provide identification and limit the ability of the untrusted computer by restricting its functionality. For example, the smart card can be used to identify the user, and then provide signed, time-stamped certificates to the untrusted computer. These certificates would permit limited functionality for a limited amount of time. This system still requires that an amount of trust, although limited, be given to the computer. In the example of ATM fraud, if the user wants to withdraw money from her account, she must allow the ATM the ability to withdraw money for a limited time. The ATM can simply withdraw more money than the user requests, while it is able, and keep the extra money.

1.4.3 Transparency

The system that we found to most closely realize the goal is described in [13]. Protocols are given that can be used to establish a secure channel between a user and remote computer through an untrusted computer. The protocols employ a form of visual cryptography as described in [14]. In this system, messages are transmitted to the user in the form of an encrypted image that conveys no information when viewed with the naked eye. The user’s key in this case is a specific transparency with an image that is similarly useless when taken by itself. By placing the transparency over the transmitted image, the message is revealed to the user. When put in the framework described by the model in Section 1.2, the user can be sure that messages that she receives are authentic because only the proxy is able to transmit messages that would reveal a predictable message when combined with the user’s transparency. The protocols allow authentication of the user by requiring that a specific response to the decoded message be sent back to the proxy. Since the user is the only person who has the transparency, no one else is able to decode the message and send back the correct response. Note that because the user has no way to encrypt messages, her response must be sent in the clear.

This method accomplishes our goal of providing authenticated bidirectional communication between the user and proxy, and could fully prevent ATM fraud when used properly.
The bank can authenticate the user’s identity by her ability to decrypt messages, doing away with the need for a PIN that could otherwise be stolen. The bank is authenticated by transmitting images that show appropriate messages when decrypted, allowing a means for the user to confirm that her requests were received correctly by the bank.

The main drawback to the transparency scheme is that it does not provide ease-of-use, primarily due to implementation issues. First, decoding a message requires very accurate alignment of the encrypted message and the transparency. This is feasible when the message is printed on a piece of paper, but it is a very challenging task when the encrypted message is displayed on a computer screen. Also, each transparency can only be used a small number of times, requiring the user to have many transparencies in order to carry out multiple transactions. Therefore, there must be a system in place that allows the user to keep track of which transparencies have already been used and which one should be used for each message. The scheme also requires that the user obtain a new set of transparencies fairly frequently. Since ease-of-use is one of our primary goals, the transparency method is not a viable solution to the untrusted computer problem, but should be credited with developing the protocols that allow authenticated bidirectional communication through the use of visual cryptography.
Chapter 2

Device

As was assumed in the model given in Section 1.2, the user will have a device to assist in establishing an authenticated bidirectional channel with the proxy. Although there may be ways to establish the channel without a device, none are currently known. Two types of devices are considered: relaying devices and monitoring devices.

2.1 Relaying Device

For the purposes of this document, a relaying device is one that is used to relay messages between the user and proxy, through the untrusted computer. The data will pass through the untrusted computer and network in a format that is indecipherable to any party except the proxy and device. This can be accomplished through the use of shared symmetric keys, public-private keys, or other means.

A relaying device will be able to provide both authentication and privacy, the latter being a feature that was described as unattainable in Section 1.3.1 when using an untrusted computer to communicate. That statement refers only to data sent in the clear through the untrusted computer, such as typing in text or displaying images. When using a relaying device, the untrusted computer only relays previously encrypted data.

This brings us to the main disadvantage of using a relaying device as a solution to the untrusted computer problem. In most cases, the computer is not actually being used as anything more than a network access point, and so does not present the same threat as if the computer were to be used directly. A straightforward example of a a device that can be used as a relaying device is Personal Digital Assistant (PDA). The PDA could be connected
to the USB port of the computer and communicate with the proxy using a Secret Sockets Layer (SSL) enabled web browser. While this allows all of the security benefits of SSL, it forces the user to interact using the small display area and cumbersome input methods of the PDA, as opposed to the relatively large monitor, mouse, and keyboard of the untrusted computer [3].

A system that would have the same benefits of the PDA-based device, without the limited interface, is a device that displays messages using a virtual reality system. In fact, such a device can potentially have a far more natural and powerful interface than a standard computer. However, like the PDA, this relegates the untrusted computer to the role of a network access point, circumventing the problem. Another drawback is that the user cannot be expected to carry all of the necessary hardware required to use a virtual reality system.

Another form of relaying device is the transparency system described in Section 1.4.3. However, this is not a very good solution to the untrusted computer problem because issues involved with its implementation would make the system fairly difficult to use. These issues were described in more detail in Section 1.4.3. It should be noted that messages are transferred from the proxy to the user in a manner that can only be decoded with the user’s transparency, but messages from the user to the proxy are sent in the clear. This allows privacy in the former direction but not the latter.

For the reasons described above, including poor interfaces, circumvention of the problem, and general difficulty of use, a relaying device would be a poor solution to the untrusted computer problem and will not be the focus of this thesis.

### 2.2 Monitoring Device

When using a monitoring device, the user receives messages directly from the untrusted computer, while the device monitors the messages and verifies that they are authentic. In order to authenticate messages, the device and proxy must have some way to exchange information so that it cannot be affected by the untrusted computer and network.

In the following two implementations, described in Sections 2.2.1 and 2.2.2 it is assumed that the device and proxy have a shared secret that allows them to perform cryptographic operations. The device and proxy must agree upon the shared secret before attempting to use the device in the untrusted computer application. Also, the secret must be transferred
between the two using a secure channel. A simple example would be for the user to connect her device directly to the proxy, assuming she has physical access to the proxy, and then they can agree upon public-private key pairs.

The majority of information that a user can receive directly from a computer is visual, i.e., screen content. Although many computers have other means of communicating with the user, for example audio from speakers, these means generally communicate far less information than screen content, and are used far less frequently. Also, messages displayed on the screen are generally far more vital when attempting to perform secure transactions. For these reasons, the monitoring devices described below are limited to monitoring visual information.

It might seem that communicating visual information securely can be done by displaying the screen content to the user, and also transmitting the content with a Message Authentication Code (MAC) of the content to the device for verification. This is not a valid solution, because the untrusted computer can send the correct information and MAC to the device, and display a modified message to the user. Instead, the monitoring device must monitor the exact visual information that is being displayed to the user. Therefore the device must have a camera that can be used to monitor the screen. The following two sections describe two different implementations using a camera-based device.

2.2.1 Pixel Mapping Approach

All development for this approach was done by Blaise Gassend at the Massachusetts Institute of Technology. It is presented here as a comparison with the approach described in the next section whose implementation is described in Chapter 3.

In this approach an exact mapping is made between screen pixels and camera pixels. A black and white image is displayed in a window on the screen using relatively large pixels. Using a predefined mapping, every pixel in the displayed image is associated with at least one pixel of the image that the camera detects. In this way the camera-equipped device can exactly reconstruct every pixel of the displayed image. The mapping from screen pixels to camera pixels is defined during a calibration phase in which each screen pixel flashes in a specific sequence that allows the device to identify each screen pixel and associate it with at least one of the camera pixels. This approach requires that the camera remain stationary relative to the screen. If the camera moves, the mapping from screen pixels to camera pixels
will become invalid and the system will need to be recalibrated.

To allow a protocol that establishes authenticated bidirectional communication, the image that is displayed on the screen consists of the message and a small area at the bottom that is used to transmit a nonce to prevent replay attacks, a one-time password, and a MAC. A sample image is shown in Figure 2-1.

**Security Protocol**

The following protocol is based on the one given in [3]. It establishes authenticated bidirectional communication between the user and proxy.

- Authentication of messages from proxy to user:
  - The Proxy sends (information, encrypted nonce, encrypted one-time password, MAC(information, encrypted nonce, encrypted one-time password)) over the channel.
  - The device exactly reconstructs the screen content from what it sees through its camera.
  - The device checks the nonce, calculates the expected MAC and compares it with the on-screen MAC. If all checks succeed, the device shows a green light, and displays the decrypted one-time password on a small LCD display.
  - The user reads the screen content.
Authentication of messages from user to proxy:

- The user types a message on the untrusted computer’s keyboard. It is sent directly to the proxy.
- The proxy sends back a message that follows the format given above, where the information is a summary of the message that the user entered in the previous step. This message is authenticated as described above.
- The user reads the one-time password from the device’s LCD screen and types it on the untrusted computer’s keyboard. The one-time password is sent directly to the proxy.
- If the one-time password that the user sends to the proxy matches the one that was sent to the user, the proxy accepts the user’s message.

2.2.2 Proxy-Based Approach

In this approach the content displayed on the untrusted computer’s screen is authenticated by sending a picture of the screen to the proxy. The proxy will know what the screen should display, and so will be able to verify that the picture of the screen shows the correct content. The picture of the screen will not be an exact representation of the screen, due to image distortions, low camera resolution, and noise. Therefore the messages displayed on the screen must be in a format that allows the proxy to verify it from the imperfect picture. In the implementation of this type of system described in Chapter 3, the messages have been restricted to text only, allowing the proxy to perform Optical Character Recognition (OCR) on the received picture.

It is assumed that the device is equipped with a camera, as in the pixel mapping approach, as well as a link that allows communication from the device to the untrusted computer.

Security Protocol

The following protocol is based on the one given in [3]. It establishes authenticated bidirectional communication between the user and proxy.

- Authentication of messages from proxy to user:
- The Proxy sends information, in the form of an image containing text, to the untrusted computer. This image is displayed on the screen.

- The device takes a picture of the screen, and sends ("verify", picture, encrypted nonce, MAC("verify", picture, encrypted nonce)) to the untrusted computer using the local link; the message is then forwarded to the proxy.

- The proxy checks the nonce, calculates the expected MAC, and compares the expected MAC with the received MAC. If all checks pass, the proxy verifies that the text displayed on the screen is genuine by performing optical character recognition (OCR) on the received picture.

- The result of the OCR is compared with the information from which the image was formed in the first step. If this check passes, the proxy sends ("yes", encrypted nonce, MAC("yes", encrypted nonce)) to the device. The device checks the nonce and MAC. If all checks pass the device lights a green light.

- **Authentication of messages from user to proxy:**

  - The user types a message on the untrusted computer’s keyboard. It is sent directly to the proxy.

  - The proxy sends back an image that summarizes the message that the user entered in the previous step. The user verifies that the summary is correct.

  - The device takes a picture of the screen, and sends ("accept", picture, encrypted nonce, MAC("accept", picture, encrypted nonce)) to the proxy. The proxy verifies the message as described above. If the tests pass, the proxy considers the picture to be approved, as only the user’s device could have produced the MAC. Note that a race condition can exist in which the untrusted computer changes the content of the screen after the user has pressed the accept button on her device, but before the camera has actually taken the picture. To avoid this condition, the implementation should provide two separate buttons: one for capturing the image and one for sending the image to the proxy.
2.2.3 Evaluation and Comparison of Approaches

Both of these approaches achieve the three goals presented in Section 1.3: authenticated bidirectional communication, functionality, and ease-of-use. There are a number of tradeoffs between these two approaches regarding the last two goals.

A big difference between the two approaches is that the pixel mapping approach requires a calibration phase and that the camera remain stationary with respect to the screen for the entire session. The proxy-based approach does not have these potential inconveniences, but it pays for that advantage with the amount of time it takes to verify a message. In the proxy based approach, the picture must be sent to the proxy for verification. This will take much longer than when using the pixel mapping approach, in which the verification takes place on the device.

The pixel mapping approach has the clear advantage that it can accommodate any message, while the current implementation of the proxy-based approach is limited to text messages. It can be expected that most transactions that the user will wish to perform using the untrusted computer can be performed using text-only methods; however, the ability to authenticate any message is still highly desirable. When limiting the messages to text, the proxy based approach can authenticate a longer text message than the pixel mapping approach, assuming equal camera resolution for both methods. This is due to the fact that in the pixel mapping approach, a portion of the screen is used for transmitting the one-time password, nonce, and MAC.

An advantage that the proxy based approach has is that its message is displayed in a more pleasant manner. When using the pixel mapping approach, each pixel of the image must be large enough for the camera to identify, resulting in a somewhat blocky image, as seen in Figure 2-1. The images used in the proxy based approach can use much smaller pixels, resulting in a more pleasant image. An example is shown in Figure 2-2.

Neither approach is obviously superior to the other. Each has advantages and disadvantages that need to be considered when deciding which approach is more suitable for a given application on the untrusted computer. The pixel mapping approach is more suited to applications where every message must be verified, or in which the ability to authenticate non-text messages is required. The proxy based approach is more suited to applications in which the authentication of a text message every few seconds is adequate. The next chapter
Figure 2-2: This is a sample image that is displayed in the proxy based approach.

goes into the details of the proxy based method’s implementation.
Chapter 3

Implementation

This chapter describes a simple version of a proxy based monitoring device that was implemented in order to test the ideas presented in the previous chapters. In this prototype, a number of simplifications were made that would not necessarily be made when designing and building a system to be put into commercial use.

3.1 Overview

There are three pieces of equipment that must be considered in the implementation of the system: the device, the untrusted computer, and the proxy. In order to speed development we used an iPAQ instead of building a custom device. An iPAQ is a hand-held device made by Compaq that can be equipped with a simple camera capable of a resolution of 640 by 480 pixels. The iPAQ is much more computationally powerful than the device that would be required for this application, but it was chosen because building a custom device would have a large development cost and would not contribute much to the evaluation of the proxy based approach as a solution to the untrusted computer problem. The iPAQ was equipped with an 802.11 wireless card, allowing easy transmission of data between the iPAQ and untrusted computer. If a custom device were to be built, it would not use a wireless card, due to the expense and high power consumption. An infrared link to the untrusted computer would be a more suitable choice.

Both the untrusted computer and proxy were implemented using separate personal computers running Windows 2000, connected through a local network. The software running on the untrusted computer simply has to display the images sent from the proxy and relay
data from the device to the proxy. The proxy will have software that must be able to authenticate the message contained in the picture of the screen that it receives from the untrusted computer. Since the proxy software was developed especially for this application, that will be focus of the remainder of this chapter. The proxy software was written using a combination of C++ and MATLAB.

3.2 Image Estimation

The picture of the untrusted computer’s screen will show a distorted version of the message that is to be verified. A sample screen picture is shown in Figure 3-1. In order for the proxy to be able to accurately perform OCR on the image, the window containing the message must be restored to look as much like the original image as possible. There are a number of distortions that need to be corrected to estimate the original image. For the discussion of these distortions, the image that is displayed on the untrusted computer’s screen will be referred to as the original image. Also, the portion of the screen picture that corresponds to the original image will be referred to as the distorted image.

There are four types of distortions that can be grouped together, and are collectively called an affine distortion. The four individual distortions are scaling, translation, rotation, and shearing. The transformation required to undo an affine distortion is a linear transformation. Another distortion that can be modeled as a linear transformation is a perspective distortion, which results from the camera taking a picture from some nonperpendicular angle relative to the screen. This makes the distorted image seem as though it is leaning back away from the camera. An affine and perspective distortion are collectively called a projective distortion, which is represented by a linear transformation, as described in Section 3.2.3.

In order to perform the transformation required to undo the perspective distortion, there must be four points whose locations are known both in the distorted and original images. After some experimentation it was found that the most effective way to locate these four points was to put an easily identifiable border around the message. The border that was chosen was a simple red box, whose four corners were found and used as the four reference points. Determining the location of the box is described in detail in Section 3.2.2.

Another type of distortion that must be corrected is the distortion caused by the camera
Figure 3-1: Sample screen picture taken by the device. Note the lens and projective distortions.

camera on the device will be relatively low-quality, as compared with most commercially available digital cameras. The quality of the device's camera should be assumed to be roughly equivalent to that of a webcam. The property of a camera of this quality that is relevant to this section is that it causes a distortion that tends to warp the image so that
Figure 3-2: Demonstration of Radial Distortion. The image on the left is the original and the image on the right is the original after having undergone radially symmetric distortion. It seems to curve at the edges of the image. This distortion was modeled as the simple one parameter radially symmetric model, described in [15]. An example is shown in Figure 3-2.

The one-parameter radially symmetric distortion is defined as follows, where \((x, y)\) is the location of a pixel in the original image, \((u, v)\) is the location of the same pixel in the distorted image, and \(k\) is a constant that can be varied to accommodate radial distortions of different severity. The coordinates \((x, y)\) and \((u, v)\) are defined from the center of the image.

\[
\begin{align*}
    u &= x(1 + kr^2) \\
    v &= y(1 + kr^2) \\
    r^2 &= x^2 + y^2
\end{align*}
\]

First the size of the undistorted image is determined, based on the size of the distorted image and the amount of distortion. Then, for every pixel location in the undistorted image, whose location is specified by \((x, y)\), the location of the pixel in the distorted image is found, specified by \((u, v)\) using the equations above. It is important to note that the pixel locations are defined as integers, but the values of \((u, v)\) that are found can be any rational number. Therefore, the value of the pixel at \((x, y)\) in the undistorted image is determined by finding the weighted average of the four pixels surrounding location \((u, v)\) in the distorted image. The weights are based on the distance from \((u, v)\) to the center of each pixel, with
Figure 3-3: Screen picture before (top) and after (bottom) correction for lens distortion. The effect is most noticeable in the corners of the image.

larger weights for shorter distances. This technique is called bilinear interpolation, and is described in detail in [16].

The value of $k$ is dependent on the camera, so some experimentation was done to find the appropriate value for the iPAQ’s camera. Figure 3-3 shows an example screen picture before and after radial distortion was corrected.

3.2.2 Border Identification

The purpose of placing an easily identifiable border around the image is two-fold. First, the proxy will receive a picture of the screen that contains the message window, and the proxy will need to determine where in the picture the window is located. Once the proxy finds the
border it is guaranteed that the message is entirely contained within that border. Second, in order to attempt to correct for the projective distortion there must be four points whose locations are known in both the original and distorted images. Once the border is found, its four corners will be used as the four reference points.

The choice of the border was a major component in the design of this system. The proxy must be able to identify it easily and robustly. Also, it should not distract the user from the message. The border was chosen as a simple red box because it satisfies these requirements. The red pixels of the box allow it to be easily separated from the background. The four straight lines that compose the box can then be found using a Hough Transform, and the intersection of these lines result in the four points that are necessary to perform a projective transformation. These steps are described in greater detail below. The other requirement of the border is that it does not draw the user’s attention away from the message. We found that because the border is so simple it does not distract the user. In fact, it was found that the red border makes the window containing the message stand out from the background, which draws the user’s attention to the message.

Isolation of Border Pixels

In order to more easily identify the border, it was colored a shade of red. When processing the picture of the screen all pixels whose red, green, and blue components fall within a certain range are determined to be potential border pixels. This task was slightly more difficult than originally planned, due to the fact that colors are not displayed consistently across computer screens. Also if the lighting conditions are not consistent over the computer’s screen, the picture of the screen will show a varying shade of color. Therefore, the range of pixel values that are accepted as potential border pixels had to be large enough to accommodate this variability, but also small enough to effectively isolate the border from the background. Through some experimentation a balance was reached that accomplishes both of these requirements. An example of an image showing all pixels found to be potential border pixels is shown in Figure 3-4.

Finding Lines - The Hough Transform

Once the border is isolated from the background it still needs to be precisely located within the screen picture. Since the border consists of four straight lines, we can simplify the
task of finding the border to the task of detecting straight lines in the picture. The task of line detection is not a new one, and was performed using the Hough Transform (HT). The HT has become the most widely used method of line detection since its introduction in 1962. A simple description of the HT is that it is a two-dimensional accumulator array, where every coordinate in the array represents a potential line in the image. The value at each coordinate represents the number of pixels that fall on the line associated with that coordinate. This allows the detection of lines by finding peaks in the HT [4].

After the four lines that make up the border are found using the Hough Transform, the intersections of these lines are determined to indicate the four corners of the border. These four points can then be used to perform a projective transformation.

### 3.2.3 Projective Distortion

A projective distortion encompasses both affine and perspective distortions, which are both the result of a linear transformation. Our goal is to perform the inverse of the transformation in order to recreate an accurate estimate of the original image. In other words, we would like to make the message shown in Figure 3-1 fit into a square of known size. The linear equations governing a projective transformation are shown below, where \((x, y)\) is the location
of a pixel in the original image and \((u, v)\) is the location of the same pixel in the distorted image.

\[
\begin{bmatrix}
u' 
v' 
w'
\end{bmatrix} = \begin{bmatrix} x & y & 1 
\end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} 
a_{21} & a_{22} & a_{23} 
a_{31} & a_{32} & 1 
\end{bmatrix}
\]

\[
u = \frac{u'}{w'}
\]

\[
v = \frac{v'}{w'}
\]

There are eight independent values of \(a_{ij}\) in the \(3 \times 3\) matrix, which can be determined if the locations of four points are known in both the distorted and original image. The four corners of the message’s border in the distorted image are mapped to their corresponding corners in the original image, allowing us to find the matrix coefficients and determine the projective transformation necessary to create an estimate of the original image [16].

The size of the undistorted image is set to be the same size as the original image. For every pixel location in the undistorted image, whose location is specified by \((x, y)\), the location of the pixel in the distorted image is found, specified by \((u, v)\), using the projective transformation. Just as when performing the radially symmetric transformation, the \((u, v)\) values will not be integers, in general, so bilinear interpolation must be used to determine the value that should be assigned to the pixel at \((x, y)\). An example of an undistorted image is shown in Figure 3-5.

Performing the projective transformation is the last step in the image restoration process. At this point, a form of OCR must be performed on the undistorted image in order to determine the message contained in the image. The specifics of the OCR that was developed are explained below.

### 3.3 Optical Character Verification

After an estimate of the original image has been obtained from the distorted image, the proxy must then verify that the text contained in the image is correct. To do this, a specific version of optical character recognition (OCR) was developed for this application. The
field of OCR is not a new one, and has applications ranging from industrial processing to document scanning [12]. Traditionally in OCR applications, the task is to determine the identity of unknown characters. The problem we are trying to solve is slightly different. Instead of attempting to determine the identity of an unknown character, we are trying to verify, with certainty, the identity of a known character. This field is known as optical character verification (OCV), and is less commonly used than OCR. Its primary application is in verifying the quality of machine-printed text. The OCV system that was designed for the untrusted computer problem is described in the following sections.

3.3.1 Requirements

It is useful to first describe the requirements that will be demanded of our OCV system. The primary requirement is that it must not allow false positives. This means that if the character in question is not the character that it should be, then it must be rejected. This requirement is necessary because the application of the OCV is the verification of text for security purposes. If it were possible for the untrusted computer to change the text in a manner that would still be accepted by the proxy, the system would be insecure and we
could not establish authenticated communication. The second requirement of our system is
that it be able to verify text quickly, so that responses can be returned to the user within
a few seconds.

3.3.2 Template Matching

The methods used in OCR and OCV applications can be broken into two broad categories:
template matching and feature extraction. Template matching involves comparing an un-
known character with character templates. This requires that the characters fit a known
structure, because the templates are fixed. This is a limitation that prevents template
matching in many applications, such as handwriting recognition, where there is no rigid
structure to the characters. Feature extraction is more suited to these types of applica-
tions, because it allows for some degree of variability in the characters’ structure. As its
name suggests, feature extraction methods recognize characters by extracting their unique
features. For example, the capital letter H consists of two straight vertical lines, joined in
the middle by a straight horizontal line. This type of character recognition is very flexible,
as it allows for multiple fonts [6].

We chose a template matching strategy for our OCV system for a number of reasons.
First, since the proxy will only need to perform OCV on text that it generates, it does not
need to allow for unknown fonts. Second, the flexibility that feature extraction affords would
be a poor quality for our system to exhibit, since it would allow for a greater probability
of false positives. By enforcing the strict structure required when using template matching,
we make it difficult for the untrusted computer to change a character without the change
being detected. Lastly, template matching is a faster technique than feature extraction.

3.3.3 Constraints

A number of constraints were placed on the characters to more easily enable reliable char-
acter verification. We limited the fonts to a specific size and type. The font type used is
OCR-B, which is a freely available font that was specifically developed for OCR applica-
tions. OCR-A is another font type that was developed specifically for OCR applications,
but is less visually appealing than OCR-B. In order to reduce development time, we limited
ourselves to using only upper-case letters, but the system could be extended to lower-case
and other characters, by developing templates for those characters.
In order for the OCV system to reliably verify characters, they must be large enough to ensure that the low-resolution camera can capture enough detail to be able to distinguish one character from another. The implemented system was able to verify a window consisting of 7 rows of 11 characters, resulting in a maximum message length of 77 characters.

### 3.3.4 Segmentation

Segmentation in OCR applications refers to the identification of individual characters from an image containing many characters, and possibly other objects as well. Four commonly used methods of segmentation are connected component labeling, X-Y-tree decomposition, run-length smearing, and Hough Transform [6]. These methods all assume no prior information about the location of characters in the image, however in our application, the rough location of characters is known because the layout and message contained in the original image are predefined. The precision to which the characters can be located is dependant on the accuracy of the image restoration. This prior knowledge allows us to avoid some of the complexities that arise in the four common techniques.

Since the image restoration will not be exact, the precise location of characters is not possible. Instead, a window is defined that is slightly larger than the size of a character. When searching for a given character whose center is at location \((x, y)\) in the original image, the window is centered at \((x, y)\) in the reconstructed image. Assuming the reconstructed image is a reasonable estimate of the original, the character will be roughly centered within the window. The portion of the image contained in the window was then thresholded to create a binary image. This form of local thresholding was performed, instead of thresholding the entire image, in order to accommodate varying lighting conditions across the image. An example of a thresholded character within a window is shown in Figure 3-6.

Since the character is not located precisely at the center of the window, a method of shift-invariant template matching, called cross-correlation [11] was investigated. This method does not create a very strong distinction between characters, but is accurate in the sense that, given an unknown character, it can determine the identity of the character by testing its cross-correlation with every character in the set and taking the maximum as an indicator of character identity. This result is desirable for applications such as document image analysis, where the closest match for an unknown character must be found. However, for our application where a known character must be verified with certainty, a stronger
distinction between characters must be found.

Instead of using a shift-invariant form of template matching, the character is precisely located in the window under the assumption that some portion of the characters' bounding box is located at the center of the window. Therefore, the sum of the column at the center of the window will be some non-zero value, assuming that white pixels have a value of one and black pixels have a value of zero. The horizontal bounds of the character are found by scanning outward from the center, until the sum of the current column is zero, indicating a completely black column. The same strategy is used to find the vertical bounds. The following section describes a method of template matching that was developed in order to create a very strong distinction between characters.

3.3.5 Templates Used

After the character has been isolated from the background, it must be compared with a template of that character. This section discusses the choice of templates used. For all sets of templates discussed in this section, the comparison between the character in question and template was performed by simply overlaying the two and identifying matching pixels. The first set of templates that were experimented with is very simple. The templates are binary images showing a character in white on a black background. The measure of similarity between the character and the template was defined as simply the number of pixels whose
values were the same between the two. Just as in the cross-correlation method, this method led to results that could be used to identify a character by choosing the template with the largest number of matching pixels amongst all possible templates. Again, this is not the goal that we are trying to achieve, so this method is not acceptable. Various methods of applying binary templates were tested, such as assigning different weights to matching white pixels and matching black pixels. Some improvements could be seen, but the level of differentiation between characters was still not strong enough to reliably verify the identity of every character. A major reason for the failure of binary templates is that they do not allow for distortions in the detected character. If a character is slightly distorted, it will not match its template exactly, and therefore lead to lower levels of similarity.

To permit slight distortions in the characters, new templates were designed that contain four distinct areas. For an example, see Figure 3-7. Referring to this figure, the light gray and white pixels represent the character pixels that should be white, and all other pixels represent the character pixels that should be black. The dark gray area was defined to allow slight distortions in the detected character. The white section was defined to indicate a region that should contain white pixels in the detected character, regardless of distortion. Similarly, the black region in the template was defined to indicate pixels that should be black in the detected character, regardless of distortion. The measure of similarity between

Figure 3-7: Sample template showing four distinct regions.
<table>
<thead>
<tr>
<th>Character Value</th>
<th>Template Value</th>
<th>Points Assigned</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>Black</td>
<td>1</td>
<td>strong match</td>
</tr>
<tr>
<td>Black</td>
<td>Dark Gray</td>
<td>1</td>
<td>weak match</td>
</tr>
<tr>
<td>Black</td>
<td>Light Gray</td>
<td>-1</td>
<td>weak mismatch</td>
</tr>
<tr>
<td>Black</td>
<td>White</td>
<td>-50</td>
<td>strong mismatch</td>
</tr>
<tr>
<td>White</td>
<td>Black</td>
<td>-50</td>
<td>strong mismatch</td>
</tr>
<tr>
<td>White</td>
<td>Dark Gray</td>
<td>-1</td>
<td>weak mismatch</td>
</tr>
<tr>
<td>White</td>
<td>Light Gray</td>
<td>1</td>
<td>weak match</td>
</tr>
<tr>
<td>White</td>
<td>White</td>
<td>1</td>
<td>strong match</td>
</tr>
</tbody>
</table>

Table 3.1: Point Assignment for Template Matching Combinations.

the detected character and the template was defined using a point-based system. The pixel value of the detected character was compared with the corresponding pixel value in the template, and a number of points were given or taken away from an accumulator depending on the combination of values. The sum of all points that were given or taken away was used as the measure of similarity. There are eight possible combinations of pixel values, since there are two possible values in the detected character, and four possible values in the template. Some experimentation was done to determine the number of points that should be assigned for each combination in order to achieve a large distinction between characters. Each of the eight combinations fit into one of four categories. These categories were defined to facilitate the discussion of this form of template matching. The point value and category for each combination are shown in Table 3.1.

This system leads to small rewards for pixels with either a weak or strong match, small penalties for pixels with weak mismatch, and very strong penalties for pixels with strong mismatch. This means that any white character pixel at the same location as a black template pixel will result in a severe penalty, as will any black character pixel at the same location as a white template pixel. If a detected character is being compared with any template other than its own, there will be enough strongly mismatching pixels to reduce the total score significantly, indicating that the character does not match the template.

This system has the desired property that the probability of a false positive, as described in Section 3.3.1, is extremely small. The cost of obtaining this property is that the probability of rejecting a correct character is higher than it would be if the system did not have this property. The rejection of a correct character will only occur if the distortion of the
character is sufficiently high. The image restoration techniques described in Section 3.2 are accurate enough to prevent correct character rejection in almost all cases. The verification time for each character is very small, since each character need only be compared with only its own template, as opposed to comparing it with every character in the set. Therefore this system meets both of the requirements specified in Section 3.3.1.

3.4 Results and Evaluation

The success of this system is based primarily on the success of the message verification software running on the proxy. The success will be evaluated on the ability of the software to distinguish between a given character and every other character in the set. If there is a large distinction between characters, any change in the message will be detected.

A test was performed in order to measure the level of distinction between characters. The character set used for this evaluation was all letters from A through Z as well as all numbers from 0 through 9, resulting in a total of 36 characters. An image containing all characters in the set was displayed on a 17 inch flat panel. Seven pictures were taken of the screen, each from a different angle relative to the screen. To facilitate testing, the iPAQ camera was not used. The results are still valid, however, because the camera that was used has the same resolution as the iPAQ camera and both are made by Philips. For each picture the following steps were taken.

1. An estimate of the original image was found using the steps described in Section 3.2.

2. For every character in the message, the corresponding character was found in the image estimate. The detected character was then compared with a template for every character in the set and the number of points acquired for every comparison was determined.

3. For each character, the maximum number of points acquired amongst all templates was recorded. This maximum always corresponded to the template of the character that was under test. The maximum point total will be referred to as the reference total. The second largest point total amongst all templates was also recorded for each character, and will be referred to as the secondary total.
Figure 3-8: Distinction Between Characters. Each pair of bars represents an individual character, and the points obtained in a comparison with that character's template as well as the largest number of points obtained in a comparison with all other templates.

For a given character, the difference between the reference and secondary totals corresponds to the minimum level of differentiation between the given character and any other character in the set. For every character, reference and secondary totals were averaged across all seven pictures, resulting in an average level of distinctness for each character. The results are shown in Figure 3-8, with each pair of bars corresponding to a specific character.

This chart shows that the difference in point total between a character and any other character is quite large. In fact, there are only three characters where the closest character has a positive point total. Only one of these three has has a point total approaching half of the point total or the correct character. The large differences are a direct result of the heavy penalties assigned for strongly mismatching pixels, as described in Section 3.3.5. It is clear from this chart that there is a very large differentiation between characters, and that any change in the message will easily be identified, thus accomplishing our goal.
The majority of the image verification code for the prototype was written in MATLAB because of the fast development time that it allows. MATLAB is an interpreted language, so it generally results in much a slower execution time than a compiled language, such as C. This leads to an inflated estimate for the amount of time required to verify each message. The MATLAB code requires approximately one minute to verify a message, but it is expected that the task can be completed in under three seconds if written in a C or another compiled language.

It should be noted that a false positive did not occur in any of the tests that were performed. False negatives only occurred when there was a significant distortion in the image, resulting from the picture being taken from a large angle relative to the center of the screen. In the majority of these cases, only the character that was furthest from the camera was falsely rejected. For example, if the picture was taken from the bottom left of the screen, the character at the top right of the message has the highest probability of false rejection. The characters that are further from the camera are more susceptible to false rejection because they are the most distorted and severely undersampled characters in the message.
Chapter 4

Conclusion

When considering the rise in both availability of public computers and the amount of internet traffic that requires some amount of security, the ability to use an untrusted public computer to perform secure transactions becomes necessary. Our goal was to design a system that would allow the use of an untrusted computer by providing authenticated bidirectional communication, functionality, and ease of use.

After evaluating a number of different types of systems that could accomplish this goal, a specific system was implemented that employs a camera based monitoring device using a proxy based approach. The two main challenges of this approach were to accurately and efficiently estimate the original image from a picture of the image on a screen, and to perform optical character verification on the estimated image. The original image was estimated by performing both a radially symmetric and projective transformation on the image. The OCV routine was designed to prevent the acceptance of false characters. The implemented system performs reliably, in that there is a large distinction between differing characters.

There are two major limitations of the current implementation of the system. The system can only authenticate text-only messages, and each message must consist of no more than 77 characters. Also, because the device must send the picture of the screen to the proxy, the amount of time required to verify a message is dependent on unpredictable network latency. We can expect to see improvement in both of these areas, as camera resolution and the amount of computation available on hand-held devices increases. Better camera resolution would allow greater detail in the picture of the screen which would lead
to the ability to verify smaller characters, and even general non-text based images. A device with more computation power will allow image verification to be done on the device and not the proxy. The message would be displayed on the screen, and a MAC of the message would be transmitted to the device, allowing the device to verify the content without further need to contact the proxy.

Future work can be done in an attempt to combine the best features of the pixel mapping approach, described in Section 2.2.1, with the best features of the implemented proxy-based approach. Relatively large screen pixels could be used to display a message, which would not be limited to text, and the border identification and image estimation techniques of the proxy based approach could be used to reconstruct the image. This technique would not require the calibration of the pixel mapping approach, or need to send the picture of the screen back to the proxy, as in the proxy based approach. The link between the untrusted computer and device that was used in the proxy based approach can be used to transmit extra information from the proxy to device. This would remove the need to display the nonce, one-time password, and MAC that are displayed at the bottom of the screen in the pixel mapping approach.

Enabling the use of an untrusted computer to perform secure transactions is an interesting and challenging problem. The implemented camera based device provides a solution to the problem by verifying the screen content of the untrusted computer. While the implemented system has the limitations stated above, but it does provide an initial attempt that can be improved.
Bibliography


