Verification of Secure Biometric Authentication Protocols

by

Anongporn Salaiwarakul

A thesis submitted
to The University of Birmingham
for the degree of Doctor of Philosophy

School of Computer Science
The University of Birmingham
Birmingham B15 2TT
United Kingdom

June 2010

UNIVERSITY OF BIRMINGHAM

University of Birmingham Research Archive

e-theses repository

This unpublished thesis/dissertation is copyright of the author and/or third parties. The intellectual property rights of the author or third parties in respect of this work are as defined by The Copyright Designs and Patents Act 1988 or as modified by any successor legislation.

Any use made of information contained in this thesis/dissertation must be in accordance with that legislation and must be properly acknowledged. Further distribution or reproduction in any format is prohibited without the permission of the copyright holder.

Abstract

The thesis presents verification of biometric authentication protocols. ProVerif is used as the verification tool for verifying and analysing the protocols. The protocol are analysed in ProVerif model. Various attacks to the protocols are generated in order to verify whether the protocols hold their intended properties.

We have selected three biometric authentication protocols and proposed a remote biometric authentication protocol for on-line banking. Each of which has different intended purposes and properties. The first protocol is generic authentication using biometric data. This protocol provides three properties of the protocol: *effectiveness*, *correctness*, and *privacy of biometric data*. In addition, the protocol is clarified in order to verify the property of *effectiveness*. Details in chapter 3 show that without this clarification, the property of *effectiveness* would not hold.

The second protocol is a biometric authentication protocol for a signature creation application. This is a specific purpose protocol that requires successfully biometric authentication in order to proceed the user's request, signing a document. The two properties of the protocol are verified: *privacy of biometric data* and *intensional authentication*. This protocol is used for signing a document using a user's private key. Hence, extension of the protocol is required so that the *intensional authentication* property can be verified. This property demonstrates that the legitimate user signs only the document that he intends to sign. A detailed description of this work can be found in chapter

4.

The thesis further considers a remote biometric authentication protocol. Chapter 5 presents the protocol and verification of its desirable properties. This chapter shows analysis of the two properties of the protocol: *privacy of biometric data* and *authenticity*.

Next, the thesis proposes a remote biometric authentication protocol for on-line banking in chapter 6. The protocol promises three intended properties: *privacy of the biometric data*, *liveness of biometric data* and *intensional authentication*. The protocol is illustrated in detail and desirable properties of the protocol are verified.

Finally, chapter 7 concludes this study by briefly comparing properties that each protocol hold. Furthermore, we have identified the limitations of this thesis and possible areas for further research.

Acknowledgements

Many thanks go to Mark Ryan, my supervisor, who has always cheered me up and helped me a lot with my PhD studies. It has been a challenging time in the UK. I might not be a ’good supervisee’, I know. But without him, this thesis would not have been completed. Just ’thank you’ is not enough for you. You are the best supervisor.

Much gratitude is due to Eike Ritter and Georgios Theodoropoulos, my thesis group members, who dedicated their time to our thesis group meetings and gave me many wise suggestions.

To Liqun Chen, who dedicated her time to discussing the CPV02 protocol and patiently replied to my emails. The information was crucial in clarifying the protocol and ensuring success in verification.

To Stephanie Delaune, who considerably expanded my knowledge of ProVerif. Though only a brief meeting, the knowledge she gave me has helped a lot and I have drawn upon it throughout this thesis.

To Hannah Harris, for help and support. Thank you for the time you took to read this thesis and for your recommendations.

Finally, this thesis is for my family: Mum, Dad, Team, Fern, and Tom, who always support me and cheer me up. Without them, I could not have overcome the most difficult time in my life.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

User authentication is a method to verify the user's identity. User authentication can be accomplished in various different ways: by using what the user knows such as a password, what the user possesses such as a smart card, or what the user is in terms of biometrics. Traditional methods of user authentication such as passwords or smart cards authenticate the user using their knowledge or possessions. A password or a smart card can be easily stolen or given away to others. Thus, we cannot be certain whoever accesses the system is the person who has authorization. This leads to the consideration of using biometrics as an attribute to authenticate the user.

The term biometrics refers to a user's personal characteristics such as fingerprint, iris, or hand geometry and user's behavioural such as key stroke. Biometric data belongs to a particular person therefore it verifies the user by means of their personal attributes. This data is unique. It is very rare that the two biometric data are the same if they come from different persons, even twins are not the exception. As the data is the user's personal characteristics, it cannot be transferred to others and this data cannot be easily stolen or given away to others even if a particular user wishes to do so. As a result, the biometric authentication process guarantees that whoever presents the biometric data is an authentic user.

However, in some ways, the risk of using biometric data can be higher than with other methods. The accuracy of the verification is lower than with other methods. This is due to the imperfect imaging conditions (such as the fact that the user does not position his biometric data as exactly the same as when he first presents it to be stored in the system database) or the biometric features themselves are not stable (such as cuts or aging). As the user's identity must be presented along with this biometric data in order to perform the user authentication, anonymous authentication is not possible. The biometric authentication relates the user to his personal attributes therefore if the biometric data is compromised, an intruder can relate this information to the user's identity; the user's privacy is not guaranteed. Even though biometric objects such as fingers or eyes cannot be used without user's consent, the biometric characteristics such as captured fingerprints can be stolen.

A user's biometric data is in the public domain. A person will leave his fingerprint on any surface he touches or on any computer he operates. Hence, the user cannot keep his biometric data secret in the same way as he can with a password. Once the biometric data is compromised or stolen, it cannot be replaced or regenerated as other methods of authentication (such as password or smart card authentication) can. Therefore, it is very important to maintain the privacy of biometric data during authentication, as is the case with a credit card number. A credit card number is not secret since we might voluntarily cite it over the telephone or via the internet. However, we want to treat it as though it were private because we do not want such data to be spread around without restriction.

Authentication in biometric protocol can be compromised in a number of ways: via an attack on the server storing the biometric code, an interception of the biometric data when read by the biometric reader, or an attack during biometric data transmission.

Therefore, security of a biometric authentication protocol is especially important because biometric data are not easily replaceable as passwords or tokens if compromised or lost. Furthermore, it is vital to be able to verify the security property effectively to show what it holds.

For example, the Needham Schroeder authentication protocol was developed in 1978 and was believed to satisfy the property of security until 1995, when it was found to be susceptible to attack by Gavin Lowe [22].

Due to the increasing use of biometric data for authentication, both instead of and alongside traditional methods, development of robust biometric authentication protocols increasingly merits consideration in terms of research.

One of the major considerations when using biometric data is its nature in which the biometric data is in the public domain. An intruder can capture the biometric data which is left on the surface the user touches and later use it to authenticate to the system as a legitimate user. Hence, the biometric authentication works well if the verifier can prove that the biometric data comes from the live presentation of the user at the time of verification.

The security level of using biometric data lessens when it is used repeatedly in various applications as a password is used. An intruder has more chances to learn the password; as in the case of biometric data. However, the risk is much higher. If a password is compromised, a user can change it easily. This case cannot be applied to the biometric data. Once the data is compromised, it is compromised for life. Thus, it is very important that the user is confident with the security requirement provided by the system he wishes to give his biometric data to.

Even though the biometric data cannot be kept secret, its privacy is desirable. The biometric reader which is involved in the authentication protocol must be guaranteed that it does not manipulate the data with an intruder. When the biometric data is released during authentication, the application which uses this data should not distribute to other agents which are not involved in the authentication process. Therefore, the verification and analysis of the biometric authentication protocol are different from the traditional protocol, such as password-based authentication protocol, that intends to keep the authentication token secret.

Approach for the verification of the password-based authentication protocol and biometric authentication protocol is distinct. For the password-based, the cryptography is an important concept in order to protect the password token while the biometric authentication tries to keep its token, the biometric data, private thus the cryptography is of minor concern. However, the cryptography is still needed in the biometric protocol to protect the biometric data not to be easily exposed to an external entity as it is one of the desired properties of the security protocol. The major concern in terms of security for the biometric authentication protocol is the freshness of the biometric data. As biometric data can be captured without the user's consent easily, it is crucial to the protocol to be able to prove that the biometric data is presented by the user at the time of user authentication. As a result, the protocol design, protocol consideration and requirements are different which leads to the verification of the two protocols being diverse.

Biometric user authentication can be used in a range of applications, from logging on to a computer locally, to remote user authentication in on-line banking, for example.

Biometric user authentication is useful and efficient in supervised situations such as passport border control, but it is significantly more risky from a security point of view if it is used in non-supervised situations such as remote user authentication. For example, an imposter might use a rubber finger that replicates the real user's fingerprint.

Detection of fake biometric data often relies on measurement of physiological signs such as temperature or pulse, but that is beyond the scope of this thesis. This thesis concerns internal security problems that might occur within the computer system and biometric reader. Can an imposter capture the presented biometric data and later insert it into the communication channel during authentication? This could be overcome by using a biometric sensor contained in the trusted platform module, for example. This would therefore guarantee that the biometric data read by the biometric reader is not stolen since it has been read by the trusted biometric reader [23].

As stated earlier, biometric authentication can be used instead of or to complement other methods of authentication. Different situations or applications require different types of protocol. Applications are diverse, and include login to a local PC, remote login, or signature creation. Therefore, several different biometric authentication protocols are considered in this thesis, each of which serves a different purpose and has different requirements in order to fulfil the intended purpose.

As describe earlier it is important to ensure the components involved in the biometric authentication system should be trusted, the CPV02 [23] is chosen among other authentication protocols as it serves the trustworthiness specification in order to secure the biometric data. The research provides the biometric authentication for general purpose. Therefore, this protocol can be used if any application needs biometric verification to prior accessing a resource or system. We intend to investigate the requirements of verification and analysis of the biometric authentication protocol that uses the trusted computing concept in order to increase the security level for biometric data.

The WSE04 [7] is picked as an example of the biometric authentication protocol for specific purpose. This protocol uses cryptographic messages for security purpose. The outcome from the verification provides us with the important properties that a specific purposed biometric authentication should hold.

A remote biometric authentication, PS06 [17], is selected in order to gather the verification requirements and considerations when designing and developing a remote biometric authentication.

Table 1.1: Biometric Authentication Protocols Comparisons

Protocol	Purpose	Requirements
CPV02 [23]	Biometric authentication protocol for general application that requires biometric user authentication	The Trusted Platform Module (TPM) is used to guarantee the correctness of the components
WSE04 [7]	Biometric authentication protocol for special purpose i.e. signing a document	A document is shown via the secured viewer to prevent the forgery of the document by an attacker
PS06 [17]	Biometric authentication protocol for remote login	The Trusted Platform Module (TPM) is used to sign biometric data to guarantee the origin of the data

In order to research the verification and analysis of the biometric authentication protocols, we have chosen three different biometric authentication protocols which serve the purposes and security requirements differently among the others. We aim to show how the verification and analysis approach are different for different types of the biometric protocol and we can then conclude security requirements and verification approach in common.

Table 1.1 shows brief characterisation of the three selected protocols (the detail descriptions and illustrations are presented in the later sections). This table summarises and compares the three protocols, their respective purposes and their requirements. As shown, different intended purposes dictate different sets of required properties. Therefore, the protocol verification process will differ for each of the three protocols.

Proposed biometric authentication protocols in the literature are either generic, for use in a range of applications that require biometric authentication such as CPV02 [23] or for use in specific applications, such as WSE04 [7]. In CPV02 [23], the protocol uses a trusted platform module to verify the components involved with biometric authentication. In the interest of verification and analysis the biometric authentication protocol which uses a trusted platform module, its meaning, purposes and requirements are studied and investigated. This thesis describes a detailed description of a trusted platform module in chapter 2.

We then study another biometric authentication protocol which has a specific purpose: creating a user's signature to sign a document.

We not only detail both types of biometric authentication protocol but we also model the protocols in ProVerif to verify their properties. One of the protocols needs to be clarified (details can be found in chapter 3) in order to verify one of its properties satisfactorily, and we extend part of the other protocol. Moreover, for the other protocol, part of the protocol is extended (details of this can be found in chapter 4).

This thesis shows that naive interpretation of a protocol can lead to a false verification result. The protocol description should be clearly specified to avoid this type of verification failure.

The thesis also describes a remote biometric authentication protocol PS06 [17] in chapter 5. This protocol is verified and analysed. Solutions to identified flaws are also proposed. Next, in chapter 6, a new remote biometric authentication protocol for on-line banking is proposed. We define intended and required properties of the protocol, verify and analyse them.

To conclude the thesis, we summarise our results in chapter 7. We also present desirable properties that we think should hold for a biometric authentication protocol. Areas for further work are also presented.

1.1 Research Contribution

The thesis makes a number of contributions of verification and analysis of biometric authentication protocols. Details are described in later chapters.

Protocols Verification and Analysis Three different purposed biometric authentication protocols are verified and analysed of which two of them are looked at in detail. The three protocols were selected from several possible biometric authentication protocols; one is for general application that requires biometric user authentication and the second protocol for a particular purpose, a signature creation application. The third is for remote login to a remote service.

Protocol Clarification and Extension This research clarifies the first protocol in order to verify its properties. Without clarification of the protocol, verification showed that certain required properties did not hold. The second protocol is extended so that one of the proposed properties can be verified. We analyse the properties of the protocol and propose the properties that the biometric authentication should hold.

A New Remote Biometric Authentication Protocol The thesis proposes an alternative remote biometric authentication protocol. The proposed protocol uses on-line banking as an example for illustration. The established properties of the proposed protocol are modelled, verified and analysed.

Desirable Properites This study of various biometric authentication protocols leads us to propose desirable properties that should hold for any biometric authentication protocol.

1.2 Structure of the Thesis

In chapter 2, related research is presented. This involves an introduction to research on biometric authentication protocols, research about verification of security protocols, verification tools, analysis of biometric authentication protocol using UML and UMLsec and hostile use of a smart card to store biometric data. This chapter also presents the powerful attacker, Dolev Yao style attacker. We have modelled an attacker to the protocol using Dolev Yao style. Moreover, to give a better understanding of trusted platform computing, the detail is presented in this chapter.

In chapter 3, the verification of integrity and secrecy properties of a biometric authentication protocol is presented. This includes a detailed description of the protocol. Moreover, this chapter presents clarification of this protocol in order that the properties of the protocol could be verified. This chapter shows that the verification results before and after the protocol clarification are different.

In chapter 4, a biometric authentication protocol of a signature creation application is presented. This chapter includes a review of the protocol, the intended properties of the protocol, and the verification results. The extension of the protocol is presented so that a property of the protocol could be verified.

In chapter 5, a biometric authentication protocol for remote login is illustrated. A detailed description of the protocol is given and its properties are verified.

In chapter 6, a new biometric authentication protocol for remote user authentication is proposed. This chapter introduces an on-line banking system as an example for illustration.

In chapter 7, a summary of the thesis is presented. This chapter shows conclusions that can be drawn from this research. This chapter discusses limitation and proposes avenues for future work.

1.3 Publications Resulting from this Thesis

The following publications have resulted from the research presented in this thesis:

Verification of Integrity and Secrecy Properties of a Biometric Authentication Protocol. A. Salaiwarakul and M. D. Ryan. Fourth Information Security Practice and Experience Conference (ISPEC'08). LNCS, Springer,2008 [1]

Analysis of a Biometric Authentication Protocol for Signature Creation Application. A. Salaiwarakul and M.D. Ryan. Third International Workshop on Security (IWSEC 2008). LNCS, Springer, 2008. [2]

Chapter 2

Related Research

Here, a summary of related research is presented. Topics covered include basic knowledge of biometric authentication, biometric data protection, verification of security protocols, biometric authentication protocols and trusted platforms. In addition, Dolev-Yao style attackers, and possible problems with smart card use are included. Some basic definitions are also presented in this chapter.

2.1 Basic Knowledge of Biometric Authentication

Biometric authentication is an authentication method that employs the user's physiological or behavioural characteristics. Examples include fingerprint recognition, face recognition, iris recognition, hand geometry, and keystroke recognition.

In a biometric authentication protocol, the user first registers his biometric code on a system. Biometric code is normally stored using a template rather than in its raw format. During user authentication, the user's biometric data is read via a biometric reader. This data is then compared with the stored biometric code.

Processes involved in biometric authentication can be classified into two steps: enrolment and verification. In the enrolment process, the user's registered biometric code (BC) is either stored in a system or on a smart card which is kept by the user. In the verification process, the user presents his biometric data (BD) to the system so that the biometric data will be compared with the stored biometric code.

In order to generate biometric data to be used in the biometric system, the raw data is processed by a feature extraction algorithm. This process locates and encodes the distinctive features of the raw data. The template is created; it is a small file derived from the extraction. During enrolment, this template is stored as the user's biometric code. Once the user presents his biometric data, the raw data will go through these processes in order to generate the template to be matched with the stored biometric code.

Several advantages of using biometric authentication instead of other methods of authentication are that:

- Biometric data cannot be given away to others even if the owner wishes to do so.
- User biometrics cannot be stolen. Biometrics refers to the user's actual physical or behavioural characteristic such as their finger.
- Biometric authentication verifies the user based on their attributes, not on what the user has or what the user knows.
- Biometric data is unique to each person. It is rare to find the same biometric data even from thousands of people.

Despite these advantages, biometric authentication is difficult to handle in comparison with other types of authentication data:

- Biometric data could be stolen from, for example, a fingerprint left on a cup. Once it is stolen, it cannot be replaced or regenerated in the same way that a password can.

- Biometric data is not stable; it degenerates through age and injury.

Biometric data can be used in identification or verification of the user. For identification, the system matches the presented biometric data against a large set of stored biometric code. If there is a match, the user is in the system and positively identified. For verification, the user needs to present his identity, e.g. his user name, along with his biometric data. The system will compare his presented biometric data with the particular biometric code that he claims is his. If there is a match, he has been positively verified.

The result of verifying the biometric data against the stored biometric code is never a total match as would be the case with a password. This is because the exact pattern presented to the sensor will always vary slightly on each occasion.

This results in an error rate in a biometric authentication system. The error rate can be categorised in terms of *False Reject Rate* (FRR) and *False Acceptance Rate* (FAR). The false reject rate is the percentage of times that a legitimate user will be refused access to the system. The false acceptance rate is the percentage of times that an imposter will be accepted as a legitimate user. These error rates are bound to the security level of the system. It is desirable for these rates to be zero but this will never be the case. The user might seek a device that provides more accuracy. These rates are published by the manufacturer. When comparing two biometric readers, considering only one of either the FRR or the FAR is not sufficient. A biometric reader could not be considered accurate if it had a low FRR but a high FAR. However, when both of the values are published, two systems are not comparable if one has a lower FRR and higher FAR and the other has a higher FRR and lower FAR. Therefore, the EER (Equal Error Rate) is a threshold value for measurement. The EER is the point where the FRR and FAR intersect. The lower the EER,

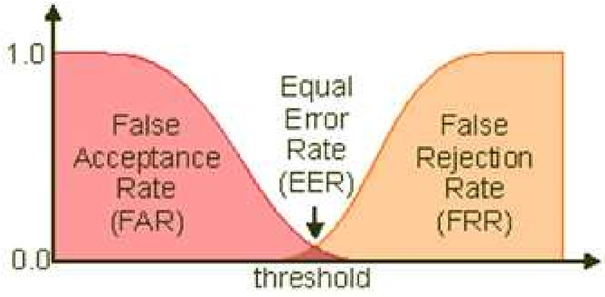

Figure 2.1: FRR, FAR , EER

the better the system performance.

To support the growth of a biometric authentication system, the system should support several biometric devices from several vendors. Therefore, this facilitates the biometric data interchange between systems and ensures interoperability of biometric data. CBEFF (Common Biometric Exchange Formats Framework) has been developed by NIST (National Institute of Standards and Technology) and the Biometric Consortium. The CBEFF defines the file format for the biometric data. This includes *header files* which specifies the biometric data format structure, *domain of use* which specifies the applications that use the biometric data such as a smart card, and the *process* that is required in order to generate the biometric data that meets the specification [32].

2.2 Biometric Data Protection

After having investigated the basic knowledge of biometric authentication including nature of biometric data, processes involved in biometric authentication shown in previous section, this section goes into detail in term of research related in biometric data protection. A review of the literature of the risk of using biometric data in user authentication and how to protect this information from a hostile attacker is presented. Various approaches that are used to protect biometric data are presented. The following describes research and approaches of protection biometric data.

As biometric data is not secret, several research papers propose ways to protect it from a malicious attacker. One of the proposed techniques involves hiding biometric data. A protocol that employs this technique is proposed by Jain and Uludag [5]. In this protocol, before biometric data is transmitted, the protocol produces a syntactic biometric data and uses it to carry out the real one by hiding the real biometric data in the syntactic biometric data. If the protocol is attacked, the attacker will believe that the obtained biometric data is the real data when in fact it is not. This paper presents a technique for hiding facial data in fingerprint data as an example.

Instead of generating fake biometric data, Ratha et al [4] suggest using the challenge and response technique to prevent resubmission of captured biometric data. They propose a biometrics-based secure authentication system. The paper outlines eight possible types of attacks on the system:

- Fake biometric data. An imposter produces fake biometric data such as a rubber finger, as described earlier.

- Reuse of captured biometric data. An attacker bypasses the biometric reader by presenting the captured biometric data direct to the system.

- Replace feature extract. Once the biometric data is read by the reader, the feature extraction process is executed in order to transform the data into information that is useful to the system. If an intruder could tamper with the extractor, the biometric image would be changed to whatever an attacker desires.

- Replace feature set of image. After the input image is extracted, the feature set of the image could be replaced if the feature extractor is installed in one place and the feature image has to be transferred to another for the matching process. An attack could occur during transmission. This could result in denial of service, preventing the legitimate user from using the system by replacing a different feature image.

- Replace matcher. This attack involves tampering with the biometric matcher to produce the desired result.
- Tampering with the stored biometric code. A stored template of the biometric code is vulnerable if it is stored in a server that has insufficient security measures in place.
- Attack the communication channel. This could compromise the system in a number of ways: by capturing transmitted biometric data, capturing a biometric matching result, or capturing a decision result.
- Decision overide. If access is gained to the decision result that allows or denies access to the system, an intruder could alter the result to allow himself access.

This research proposes a method that prevents the second type of attack from taking place: resubmission of captured biometric data to the system by bypassing the biometric reader. The biometric reader produces a challenge and response random number to be included in the biometric data that the user presents. As a result, when the system or the matcher fetches the data, the freshness and live presentation of the data can be verified. The freshness can be validated by checking that the number has not been used before. This number also guarantees the live presentation of the data since the number is generated by the biometric reader. With this number, there is no other way for an intruder to bypass the biometric reader.

Another research that Ratha et al [3] propose is a method for hiding data into biometric data. This paper creates an on-line fingerprint authentication system for commercial transactions. A different verification string is created by the service provider for each transaction in order to prevent replay attacks.

The verification string is mixed up with the biometric data before transmission. The verification string is combined with the biometric data in a different way each time so that it would not be possible for an attacker to learn how to extract the biometric data. The location of this string is different based on the structure of the image. The input image of the biometric data is decompressed, then the data-hiding algorithm is performed. Here are the four steps of the data-hiding algorithm:

- Site selection site S: This stage collects indices of all possible sites where a change in least significant is tolerable and chooses candidates.
- Random number seeding: This step selects the sites from the set S. The random seeds are calculated and picked. Randomly picking the seeds ensures that the message is embedded in different locations.
- Bit setting: The message is translated into bits.
- Bit saving: This step is optional. The original low order bits (bits that were not selected for the site) are saved and appended to the bit stream as a user comment field.

When receiving the biometric data, the recipient decompresses it and validates the verification string. The verification string is combined with the biometric data in a different way each time so it would not be possible for an attacker to learn how to extract the biometric data.

Khan and Zhang [6] present *'Implementing Templates Security in Remote Biometric Authentication Systems'*. They propose a technique for protecting biometric data that uses a secret key to encrypt biometric data. The system generates a secret key each time the user performs biometric authentication. This secret key is shared between the server and the user, and is used to encrypt the user's biometric data. Using this technique, the intercepted biometric data cannot be reused by an attacker. This proposes a One-Time Biometrics.

The proposed system comprises two processes: *secret key generator* and *encryptor and modulator*.

- Secret Key Generator: This process generates secret keys. The first one is randomly generated. The other keys (parameter, modulation, and seed keys) are generated based on this session key.
- Encryptor and Modulator: This process encrypts and modulates the biometric template in order to secure the biometric data.

After the session key is generated, it is transmitted via an SSL channel. This session key is used to generate the parameter, modulation and seed keys on the recipient side by an agreed algorithm. These keys are then used for demodulation and decryption of the received biometric data. The decrypted data is matched against the stored biometric code in the database.

2.3 Security Protocols

Different security protocols are created to achieve different goals. These security protocols can vary in terms of secure channel protocols, such as SSH or SSL, or e-voting, and biometric authentication protocol. This section describes the importance of the verification of the security protocols, tools that are widely used for verification and various approaches in verifying security protocols.

2.3.1 Verification of Security Protocols

A protocol can be verified in a number of ways, from manual to formal verification techniques such as model checking, equivalence checking and theorem proving. Research in protocol verification is a fertile area because security protocols are error prone and it is not easy to identify errors through manual verification. Therefore, automatic verification tools are useful. These tools,

Table 2.1: Verification Tools and Security Protocols

Verification Tools	Examples of Security Protocols	Reference
Isabelle/HOL	Otway-Rees, Needham-Schroeder, TLS	[38]
Avispa	TLS	[40]
Casper/FDR	Wide-mouthed-frog protocol	[44]
ProVerif	E-Voting (FOO92)	[15]
	JFK	[13]

such as Avispa, ProVerif, and Scyther are examples of well-known automatic verification tools for security protocols. Table 2.1 gives examples of some security protocols that have been verified using automatic verification tools.

In [33], six verification tools are compared: Avispa which consists of four tools, CL-Atse, OFMC, Sat-Mc, and TA4SP; ProVerif; and Scyther. The security properties of each tool are modelled. In each tool, the secrecy of nonce and session key are analysed and the performance of each tool is compared. ProVerif is shown to be the fastest tool in terms of time taken to verify the security properties.

As discussed above, security protocols are vital in biometric authentication and their correctness must be proved. As it is difficult to validate them manually, their correctness can be checked using formal verification techniques such as model checking, equivalence checking, or theorem proving.

There are various papers on this topic. In [15], an analysis of an electronic voting protocol using applied pi calculus is presented. This research considers three properties of the voting protocol: fairness, eligibility, and privacy. The authors first formalise the protocol model in applied pi calculus and then verify the first two properties using the automatic tool ProVerif, while the third property is verified using a manual proof technique.

This research verifies the FOO92 voting protocol by modelling it in applied pi calculus. This protocol is composed of a voter, an administrator, and a collector. The voter registers his intention. The administrator verifies that the vote comes from the legitimate voter. The collector collects the votes.

In the analysis section of this paper, the fairness property is analysed in order to verify that no vote is leaked before the opening ballot phase. The fairness property is modelled as a secrecy property. Another possible attack on the fairness property is a guessing attack. The fairness property tends to keep the votes secret. An attacker could try to guess votes by encrypting the guessing vote with the administrator's public key and comparing the result with the vote that is captured from the legitimate user. The verification result is positive.

The eligibility property declares that only the legitimate voter can vote and only once. This model cannot verify that the voter can vote only once because all votes share the same key.

The verification of the privacy property tends to verify that the link between the voter and his vote is hidden. The verification checks that the voter *V1* voting *Vote1* and voter *V2* voting *Vote2* is observationally equivalent to voter *V2* voting *Vote1* and voter *V1* voting *Vote2*.

Delaune et al. [16] present the first formal method of coercion-resistance and receipt-freeness. Moreover, they verify these two properties of an electronic voting protocol. Before this research, the coercion-resistance and receipt-freeness properties were in natural language and difficult to distinguish.

Automated theorem prover is one of the effective tools in order to analyse the security goals provided by a protocol. Jurjens [41] presents the code analysis using automated theorem provers. The research shows the approach to analyse the security goals by examining in source-code level. The author uses automated theorem provers (ATPs). This research applies the proposed approach to analyse a biometric authentication if it provides the intended security guarantee.

This research uses the Dolev-Yao style adversary in analysing the security goals. This style of adversary is able to read messages over the network and collect in its knowledge set. The attacker can also calculate the attack from its knowledge set. The protocol includes user, smart card, host system and a biometric sensor. This paper assumes that the attacker can somehow obtain the possession of the smart card and can access the communication channel between the smart card and the host system. If the analysis reveals that there could be an attack against the protocol, an attack generation script written in Prolog is generated from the C code.

The author describes how to transform the control flow graph generated from the C program to first-order logic, which is given as input to the automated theorem provers (detail can be found in [41]). This research does not aim to provide an automated full formal verification of C code but it rather gives a better understanding of the security properties of the protocol implementation to facilitate use in an industrial environment.

2.3.2 Security Analysis of a Biometric Authentication System

The previous section describes various security protocols, several tools, techniques and approaches that can be used for protocol verification. This section places emphasises upon approaches for the security analysis of a biometric authentication system.

Lloyd and Jurjens [43] develop a UMLsec approach for security analysis of a biometric authentication system. The research adapts a remote biometric authentication system proposed by Viti and Bistarelli [45]. They investigate the system using the Java Modelling Language (JML) and analyse the security specification in UMLsec. This research also compares advantages and disadvantages of both approaches.

The biometric authentication system is simply described as the PC is connected with a combined scanner/smart card reader. The PC is a host for authenticating the user through biometric authentication. The smart card contains a biometric template which will be matched with the scanned user's biometric data. If the biometric verification result is matched, the result and a nonce are encrypted with user's private key which is stored in the smart card. The encrypted data is sent to the server in order that the server will decrypt and check the validation of the data. This completes the process of authentication.

The research models the system requirements in UML and specifies the security requirements in UMLsec. They implement the software components of the system in JML to verify the systems code against the UMLsec specification.

Another prospective of verifying and analysing security requirements for biometric authentication is using UML. The paper [42] proposed by Jurjens presents an extensible verification framework for verifying UML model for security requirements. This paper presents an approach to translate behavioural UMLsec diagrams to formulas in first-order logic. Theses translated formulas are input into an automated theorem prover supporting the TPTP input notation. If an attack is found, an attacker generator produces an attack scenario. The protocol designer can then correct the protocol.

In order to apply the framework, the developer creates model in UML format. The dynamic checker translates the UML model into the automated theorem prover input language. The results are sent to the error analyser. The error analyser describes to the developer the problem found in the text report.

The paper describes the translation of UMLsec diagrams to first-order logic (FIL) formulas which then allows automated analysis of the diagrams using automated first-order logic theorem provers. A deployment diagrams specifies the layers of the system and the security level are input. The adversary model is generated in first-order logic in the security analysis.

2.4 Trusted Computing

One of the chosen protocols uses the trusted platform computing in order to increase the security level of the protocol thus this section describes the concept of trusted computing. Understanding the fundamentals of the trusted computing platform provides correct interpretation for verification and analysis of the protocol.

The computing platform is trusted if it always behaves in the expected manner for the intended purpose [23]. The level of trust in a computing platform varies. One computing platform could be considered to be trusted for one purpose but not for a different purpose. For example, a general computer in the office is trusted for manipulating general data but it is considered to be untrustworthy when it is used for manipulating biometric data. The level of trustworthiness is set by the system administrator. The trusted platform is a computing platform that has a trusted component in the form of built-in hardware [30]. The trusted platform is the technology developed by the Trusted Computing Group (TCG). The trusted platform guarantees that the operations it performs can be trusted. This means it behaves in the expected manner. The trusted platform must be able to measure and store the state of a component, specifically, the integrity metric. Hence, the component communicating with the platform will be able to figure out if there is any state change in the platform.

Measurement and secure storage of the trusted platform are accomplished by the *Trusted Platform Module* (TPM). Each trusted computing platform contains at least one TPM.

2.4.1 Trusted Platform Module

The TPM is usually built-in hardware which can store the status of each component in the computing platform. The TPM could be considered to be a hardware chip with added cryptographic functionality. Therefore, it can be used for device authentication. It has secure storage containing a cryptographic key to protect information. Each TPM has a unique and secret public/private key pair which is installed when it is created. This is called the *Endorsement Key* (EK). The EK is unique to a particular TPM. It is generated at the time the TPM is manufactured. The EK is taken as an identity of the TPM. Hence, to ensure user privacy, there is no command to use the EK for signing. For the purpose of platform authentication, there is an *Application Identity Key* (AIK). This signing key is used for platform authentication to the service provider. A number of AIKs can be generated inside the platform in order to sign application-specific data.

Generally, when the computing platform boots up, the TPM collects the status of various components of the platform, such as the operating system or other software such as the biometric matching algorithm software. This value is encrypted by the TPM key and stored in its secure storage, specifically the *Platform Configuration Registers* (PCRs). A third party can obtain the unforgeable state of the platform from the PCR. A measurement of other components can also be included in the PCR.

Later, if the computing platform is used in a situation requiring a high level of security, biometric data authentication, for example, the computing platform will ask the TPM to measure related components and collect the integrity value. The user or component could check status changes with the PCR. The operation will only proceed if the value satisfies the system strategy that was set up by the system administrator.

In the same system, this value varies among operations. For example, the value is higher for biometric authentication but lower when the system is used for password authentication. The required value is set up by the system administrator.

The TPM could be installed in a component in the system, in the personal computer, for example, to ensure the software process performs the correct operations. The TPM could also be installed in the biometric reader in order to prevent an intruder corrupting the device to gain the biometric data [23].

2.5 Assumptions and Considerations

This section gives description of the thesis assumption and consideration when verifying the protocols. The verification and analysis of the protocol in this thesis uses the Dolev Yao style attacker to implement the attack on the protocols in order to validate the security requirements that the protocols provide. This section also presents information of smart cards that is important in order to formulate the attack to the protocol that uses smart cards to store credential data. An example of attack to the smart card communication in order to spoof the legitimate to release his credential data is shown in this section.

2.5.1 Dolev Yao Attacker

A Dolev Yao style intruder is a classical powerful attacker. The intruder can listen to, interfere with, and regenerate messages. This intruder can manipulate and create a new message from captured messages. This includes generating a cipher text if it knows the particular key or deciphering a text if it is encrypted by the known key. The intruder can play with messages on the channel it listens to. It can impersonate a legitimate user to gain information. By modelling a Dolev Yao style attacker, protocol verification is more effective. This style of intruder can be present in a local or network connection. Proposed protocol verification techniques in the literature often employ this style of intruder [15, 16].

2.5.2 Smart Card

A smart card is a plastic card that has a chip. It can store and process data. A smart card can be used for identification such as a student ID card, authentication such as a common access control card, and data storage such as a credit card.

A smart card can be considered to be a tamper resistant device. The information stored in a smart card might be a pin number and a user's account information, as in a credit card. It can also store a user's biometric code. Its content cannot be changed without use of a formal protocol. If an attacker attempts to replace or change the stored data, the smart card can no longer be used.

Therefore, if a smart card is used for storing biometric code, this code could be considered secure when stored on the card. However, the security of biometric data cannot be guaranteed when it is transmitted and used outside the card. Biometric data can be captured during transmission even in a local or network connection.

A smart card can also be used in the biometric matching process. User verification can either be carried out within the smart card, a process called on-card matching, or in the system outside the card, known as off-card matching.

The on-card matching algorithm protects the user's stored biometric code. The biometric code is not necessarily transferred to the outside environment if using this type of matching. Even though the biometric data is not considered to be secret, the protocol should not reveal it without the user's consent. In [7], a way to protect biometric data by using an on-card matching mechanism is considered; this method is reviewed and analysed in this thesis.

2.5.3 Chip and Pin (EMV) Point-of-Sale Terminal Interceptor

The thesis considers protocols that use a smart card to store a user's biometric code. This section shows the possible threats to the smart card protocol. This principal literature is important in consideration of examining an attack to biometric authentication protocol that uses a smart card to store credential value, especially user's biometric code.

Mike Bond [10] proposes a device that intercepts data transmitted between a smart card and a smart card reader. He creates an interceptor which positions itself between the point-of-sale terminal in a shop and a chip and pin card.

EMV (Europay, Mastercard and Visa) is a protocol for debit and credit payments in Europe. It is known as Chip and Pin in the UK [31]. The interceptor does not copy the chip but it listens passively to the communication between the smart card reader and the card. It gains account information and perhaps the amount the customer wants to pay. This information could be forwarded to an eavesdropper. The customer does not realise that he is interacting with a trespassed reader. The terminal shows the correct amount that he wishes to pay, but might instruct the card to pay another larger amount.

The scenario can be illustrated thus: the legitimate user uses his card to pay for the goods. He simply inserts his card into the card reader which looks authentic but is not. This card reader is modified so that it can listen to the information the customer enters, i.e. the pin number. In a normal situation, this information is forwarded to the card issuer so that the account information can be authenticated and the procedure of deducting money from the customer's account can proceed.

However, before this action takes place, the customer's information is forwarded to an intruder who is waiting for the information. This intruder is waiting at another shop and is about to pay for goods by receiving the information from the card. He inserts a modified card that has a wired connection with the laptop that received the account information. This information is sent to the card reader in the intruder's shop and pays for his goods. The legitimate user gets his goods for free but pays for the intruder's instead.

2.6 Applied Pi Calculus and ProVerif

2.6.1 Applied Pi Calculus

This section is dedicated for the verification language and tool that are used in the verification section of the thesis. The Applied pi calculus is a language for describing concurrent processes and their interactions [25]. It is based on pi calculus, but is intended to be less pure and therefore more convenient to use. Properties of processes described in applied pi calculus can be proved by employing either manual techniques or automated tools such as ProVerif [26]. As well as reachability properties that are typical of model-checking tools, ProVerif can in some cases prove that processes are observationally equivalent [27].

To describe processes in applied pi calculus, one starts with a set of names (which are used to name communication channels or other constants), a set of variables, and a set of function symbol which will be used to define terms. In the case of security protocols, typical function symbols will include **enc** for encryption (which takes plaintext x and a key k, and returns the corresponding cipher text) and **dec** for decryption (which takes cipher text and a key k and returns the plaintext x). One can also describe equations which hold on terms constructed from the function. For example:

$$\mathbf{dec}(\mathbf{enc}(\mathrm{x},\mathrm{k}),\mathrm{k}) = \mathrm{x}$$

Terms are defined as names, variables, and function symbols applied to other terms. Terms and function symbols are sorted, and of course function symbol application must respect sorts and arities. In the applied pi calculus, one has (plain) proceses and extended processes. Plain processes are built up in a similar way to processes in the pi calculus, except that messages can contain terms (rather than just names) [25, 15].

2.6.2 ProVerif

This thesis verifies the protocols using ProVerif, a verifier based on applied pi calculus. ProVerif is a protocol verifier developed by Bruno Blanchet [11], that is able to take as input a variant of the applied pi calculus [12]. It can handle an unbounded number of sessions of the protocol and an unbounded message space. This tool has been used to prove the security properties of various protocols [13, 14, 15, 16]. It can be used to prove secrecy, authenticity and strong secrecy properties of cryptographic protocols. It can handle an unbounded number of sessions of the protocol and an unbounded message space. The keywords of this input system are `among`, `and`, `choice`, `clauses`, `data`, `elimtrue`, `else`, `equation`, `event`, `free`, `fun`, `if`, `in`, `let`, `new`, `noninterf`, `not`, `nounif`, `out`, `param`, `phase`, `putbegin`, `pred`, `private`, `process`, `query`, `reduc`, `suchthat`, `then` and `weaksecret`.

The input file consists of a list of declaration, followed by the keyword process and a process:

<declaration >*`process`<process >

The grammar of processes accepted by ProVerif is described in figure 2.2. Detailed description of the grammar accepted by ProVerif can be found here:

- `equation` <term >= <term >. `equation` M1 = M2 says that the terms M1 = M2 and in fact equal. The function symbols in the equation should be only already declared constructors. The treatment of equations is still very naive and preliminary.

P, Q, R	processes
0	null process
P |Q	parallel composition
new n; P	name restriction
new x; P	variable restriction
equation <term >= <term >	the terms M1 and M2 are in fact equal
query attacker : M	determines whether the attacker may have M.
if M = N then P else Q	conditional
event x; P	event launch
let x = M in P	replace the variable x with the term M in process P
in(M,N); P	message input
out(M,N); P	message output
!P	replica

Figure 2.2: ProVerif Grammar

- `query attacker : M` determines whether the attacker may have M. `not attacker : M` is true when M is secret.

- `if f then P else Q` This test exexutes P when the fact is true. Otherwise, it executes Q. The process if f then P is equivalent to if f then P else 0.

- `event M;P` The event command emits the event `event M`, then execute P.

- `let p = M in P` The let command executes P after matching the term M with the pattern p, and blinding the varaiable contained in p. If the term M does not match the pattern p, the process blocks.

- `in(c,p); P` The input command inputs a message on channel c, and executes P after matching the input message with p, and blinding the variable contained in p.

- `out(x,M);P` The output command outputs the message M on the channel c, then executes P.
- `!P` The replica !P executes an unbound number of copies of P in parallel : P |P |P |...

In order to verify properties of a protocol, query commands may be executed. The query 'attacker: m' is satisfied if an attacker may obtain the message m by observing the messages on public channels and by applying functions to them. The query $ev : f(x_1, \ldots, x_n) \Rightarrow ev : f'(y_1, \ldots, y_m)$ is satisfied if the event $f'(y_1, \ldots, y_m)$ must have been executed before any occurrence of the event $f(x_1, \ldots, x_n)$.

An advantage of using ProVerif as a verifier is that it models an attacker which is compliant with the Dolev-Yao model automatically. We do not need to explicitly model the attacker.

2.7 Desirable Properties for Biometric Authentication Protocol

This section identifies general concepts of desirable properties of the biometric authentication protocols. After having examined various biometric authentication protocols, this thesis proposes the properties that it is believed the protocols should hold.

- Privacy of Biometric Data. The privacy property serves the security requirement of the biometric authentication protocol by the nature of the biometric data. As mentioned, the biometric data should be considered private rather than secret. On top of that, the biometric data cannot be replaced, changed or regenerated if it is stolen or compromised as it could be with other types of authentication such as passwords or smart cards.

  The biometric data can be revealed, when received from the user, through a biometric reader. It could also be exposed during data transmission or disclosed by a corrupt machine involved in the biometric matching process. These are possible threats to the biometric data.

  The privacy property refers to the protection of the biometric data by scoping its use within the devices, components and channels that are participating in the biometric authentication process. The privacy property guarantees that the biometric data will be released only to the neat components and they shall not manipulate biometric data in order to perform the user's authentication as they are legitimate.

- Authenticity. The authenticity is a general achievement for authentication protocol; the biometric authentication protocol is not the exception. The authentication protocol should ensure that the person or thing is in fact is who or what it claims to be. In biometric authentication protocol, if the protocol achieves this property, it certifies that the protocol can protect an attacker from capturing the user's biometric data and replaying it as his own. The user is assured about the safe use of his biometric data for authentication. The authenticity property breaks when an intruder can successfully expose himself as a legitimate user to the system. For example, Alice successfully claims to the system that she is Bob.

- Effectiveness. For biometric authentication, in general, the biometric data or biometric code will be transferred to a platform for matching purposes. To increase the security requirements of the biometric data, the platform should be verified before the biometric data or biometric code is transferred to. For example, in [23], the protocol introduces the integrity metric to assure the trustworthiness of a component acquiring biometric data or biometric code. The biometric data and biometric code will not be released to the platform until its trustworthiness is verified and satisfied by the user.

  Hence, the effectiveness property analyses that the protocol provides checking that a component receives biometric data or biometric code only if its integrity metric is verified.

  Achieving the effectiveness property ensures that the biometric data or biometric code will not be released to the unworthiness entity which somehow later turns out to be an intruder. This decreases the possibility of spreading around the biometric data and biometric code without restriction.

- Correctness. The correctness property for the biometric authentication protocol can be described as when the user will not give his biometric data unless he is ensured of the trustworthiness of the biometric reader and the computing platform that operates the biometric matching process. This property is proposed due to the fact that if the biometric data is captured by an attacker, it is lost for life and his authentication token cannot be recovered. As a consequence, a biometric authentication protocol should achieve this property to guarantee the user that his biometric data will not spread around to an attacker as he recognises that his biometric data cannot be recovered if it is compromised.

The protocol should provide an approach to verify the biometric reader and the computing platform. Moreover, the result from the validation should be shown to the user so that he will judge his decision before releasing his biometric data. The trustworthiness of the biometric reader and the computing platform assures the user that it will not manipulate the data to an intruder.

In this thesis, the correctness property does not refer to the correct functions that the biometric reader or computing platform provides to the user. The correctness property in terms of security analysis for biometric authentication is shown illustrated in [23]. Therefore, even if the platform provides the correct function to the user, for example, the biometric reader can scan the biometric data template, it may not supply the correct trustworthiness to the user; the user may not trust this biometric reader as he is unsure of whether it could be tampered with by an intruder.

The correctness property is proposed to serve to verify the protocol that involved the devices that are not possessed by the user, for example, in the public internet cafe, the user may wish to authenticate himself using his biometrics via the public biometric reader and public computing platform. Therefore before the user release his biometric data to the reader and the computing platform, he should be sure of their correctness; he is satisfied with the security level of the devices.

- Liveness. From the review of the literature, a rubber finger or fake biometric data can forge the biometric system [4], [5]. Therefore, designing and verifying biometric authentication protocols should consider this threat. The biometric authentication protocol should confirm that the biometric data it is processing comes from live presentation of the user at the time verification.

  As the biometric data can be captured in public places, the biometric authentication protocol should complete this property in the interests of achieving the preliminary security. Without providing this property, the protocol could be risked from using an artificial biometric data to authenticate the user. The risk is much higher when the protocol is used in a non-supervised situation such as an on-line banking transaction that requires the biometric authentication. To provide an approach to verify and analysis the protocol and to propose a remote biometric authentication protocol that serves this property, this thesis proposes a remote biometric authentication protocol. Detailed discussion and description of the protocol are shown in chapter 6.

- Intensional Authentication. This property ensures that the protocol would not be easily tricked by an intruder to perform an action, provided by the application or system, which he does not wish to perform. In a normal situation, the user authenticates to the system using his biometrics only if he wishes to engage in the application that the system provides. In some situations, an attacker wishes to use the application on behalf of the user. He could try to manipulate the data or messages and lead the user to do whatever action he wishes without willing to do so. This property is variant to the protocols due to each protocol providing different purposes. Therefore, the provided application of each protocol is different, such as signing application offers the signing process for a document or bank applications offering bank transactions. The intensional authentication in detail is different to each protocol but the main concept is that the protocol prevents the user from being fooled by an attacker to lead him to do whatever action he does not intend to do.

One of the example situations could be illustrated as a user, Bob, authenticates to the signing application in order to perform signing a document ”B”. An intruder, Alice, tries to manipulate the messages to trick Bob to sign a document ”A” using Bob’s signature.

This property refers to the effective use of biometric data only for the purpose specified by the protocol. This property is discussed in detail in chapter 5 in relation to the signature creation application. This property guarantees that the user signs only the document that he has been shown and has agreed to sign.

This section describes the desirable security requirements as a generic abstraction, the later sections illustrate the properties of each protocols specifically to their intended purposes, consideration in terms of prospective security and components involved in the protocols.

Chapter 3

Verification of Integrity and Secrecy Properties of a Biometric Authentication Protocol

One of the case studies that are chosen for the thesis is a biometric authentication protocol which is proposed by Chen et al [23]. This protocol is a generic protocol for biometric authentication. It can be used as a protocol in applications that require authentication before the user is allowed to proceed.

After having reviewed the research (presented in section 2.2), we have discovered that most of the research tries to protect biometric data as it is secret but [23] views it differently. As shown in section 2.2, the proposed approaches try to hide the biometric data or change it to a different form before being transmitted. In contrast with the above, [23] considers the implications of the fact that biometric data is public.

The considerations of using the Trusted Platform Module (TPM) are significant in this research. The TPM is used in this protocol to guarantee the trustworthiness of the components holding biometric data and biometric code. The next section shows the detail of the protocol, how the protocol proceeds to authenticate the user using their biometrics and the use of the trusted computing in this protocol in order to increase the security level.

3.1 The CPV02 protocol

In [23], Chen, Pearson and Vamvakas present a protocol for biometric authentication that we call CPV02. The protocol considers that the biometric data is public rather than secret data such as the password. The interesting part of this protocol is that it uses TPM concept to validate the integrity of the components dealing with biometric data. All the validated (by TPM) components are trusted to process biometric data. The trusted platform module could be considered as a processor that can store the stage of the components. From this stored value, the stage of the components can be verified whether they are changed or different from the previous stage.

This protocol prevents disclosure of biometric data both during data transmission and within all system hardware. This is achieved through integrity metric checking. The TPM first checks the computing platform when it boots up. The value obtained from this check is called the integrity metric. This integrity metric is stored securely in the TPM. Any change of software or hardware triggers the TPM to check and record the integrity metric again. The user or another component can use this value to decide whether or not to trust that component to proceed with its transaction.

In this protocol, the system under consideration is composed of three connected components: a smart card (SC), a Trusted Computing Platform (TCP) and a Trusted Biometric Reader (TBR). The computing platform and the biometric reader are trusted only if their integrity metrics are satisfied. The value must be checked before the protocol communication starts.

The SC is used for storing credential information such as the user's biometric code (BC) or the user's signature. This protocol assumes that the smart card is a trusted device, it requires the computing platform to send its integrity metric so that the smart card, more precisely the user, can decide from this value whether to transfer the secret data to the platform or not. The TBR is a device for reading the user's biometric data (BD) for use later in the matching process. In this protocol, the TBR and the SC generate session keys to transfer the user's submitted biometric data (BD) and the user's stored biometric code (BC).

Generally, a TCP is a computer platform which contains at least one Trusted Platform Module (TPM). The TPM is a device that behaves in an expected manner for the intended purpose and is resistant to attacks by application software or viruses [24]. This is achieved because the TPM stores keys and can perform cryptographic operations. The TPM can check the integrity of the TCP. Specifically, it can create an unforgeable summary (integrity metric) of the software on the TCP, allowing a third party to verify that the software has not been compromised. This can be accomplished by presenting a certificate to the third party to confirm that it is communicating with a valid TPM.

Before a third party accesses the TCP, it can check the integrity metric that the TPM provided. This value is signed by the TPM so that the third party can verify its validity.

Table 3.1 summarises notations and meanings that will be used through out this chapter. Figure 3.1 shows the basic system for this model. Informally, it can be described as a user holding a smart card that contains her previously stored biometric code, e.g. fingerprint code. To authenticate herself to the system, she first inserts the smart card into a smart card reader. This triggers part of the protocol during which the integrity of the computing platform and the biometric reader are checked and the result is returned to the smart card. If the smart card is satisfied that the computing platform and biometric

Table 3.1: Notations and Meanings of [23]

Notation	Meaning
BC	User's stored biometric code
BD	User's submitted biometric data
TPM	Trusted Platform Module
TCP	Trusted Computing Platform
TBR	Trusted Biometric Reader

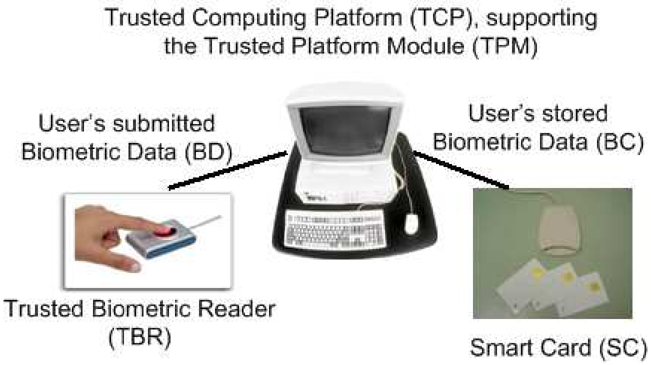
Trusted Computing Platform (TCP), supporting the Trusted Platform Module (TPM)
User's submitted Biometric Data (BD)
User's stored Biometric Data (BC)
Trusted Biometric Reader (TBR)
Smart Card (SC)

Figure 3.1: The Basic Setup for CPV02 Consists of a Trusted Biometric Reader (TBR), a Trusted Computing Platform (TCP) that Supports a Trusted Platform Module (TPM), and a Smart Card Device (SC)

reader have not been tampered with, it indicates this to the user, e.g. by releasing a special image to be displayed by the computing platform. The user recognises that image as an indication that the integrity checks have been successful and proceeds to the second step, which is biometric authentication. To achieve that, she submits her biometric data, e.g. by placing her fingerprint on a biometric reader. The biometric code stored on the smart card and the submitted biometric data from the biometric reader are then sent to the computing platform, which will validate whether they match. If they match, the smart card will release the user's credential data, e.g. her signature on a message, to the computing platform.

The biometric code is stored in the smart card and will be transferred to the TPM for comparison with the biometric data. However, before this transmission is performed, the TPM and the SC must authenticate each other by sending an authentication message, which includes a nonce and integrity metric. The integrity metric is a measurement of the trustworthiness of the component. Depending on its policy, the challenger will decide, based on this value, whether to trust or allow any action to be performed.

The message sequence of this protocol is shown in figure 3.2. It could be described as when the user inserts the smart card into the reader in order to start the user authentication; this triggers the SC to identify itself to the trusted computing platform. Then the SC sends a nonce n1 and its identity to the TPM. In response, the TPM generates a nonce n2 and a message including n1, n2, the identity of the component that the TPM wants to communicate with, in this case SC, and integrity metric D3. The integrity metric D3 is the integrity value of the trusted computing platform. The SC decides on this value whether it will continue in communication with this component or not. The message sent back to the SC is signed by the TPM so that the SC can check its origin and the correctness of n1. If the SC is satisfied with the integrity value D3, the SC generates the session key SK1, shared by the SC and the TPM, for encrypting the BC, before sending it together with the authentication messages. After the TPM has verified the message, it then stores the BC.

When the TBR is presented to the system, it also performs mutual authentication with the TPM and generates a session key to share between the TBR and the TPM. In the same way as that in which the TPM and the SC authenticated each other, the TBR sends an integrity metric D7 to the TPM. If the TPM has successfully verified the message it receives, it will send back a message MF5. The TBR verifies the message. After the authentication has succeeded, the TBR generates a session key SK2, shared by the TBR and the TPM, for use in encrypting the BD from the TBR to the TPM.

The BD is encrypted by using the session key created in the previous stage to the TPM. This data will be compared with the BC. After the message is verified, the TPM decrypts the encrypted message and verifies the validity of the BD. If they match, the user is allowed to use the system or perform the request. For example, the SC releases the user's signature.

The specification of the protocol shown in figure 3.2 uses the following notations:

$\texttt{S}_x\texttt{(m)}$ a signature of the element m signed with a private key of x.

$\texttt{E}_x\texttt{(m)}$ an element m is encrypted by the public key of x.

$\texttt{E}'_x\texttt{(m)}$ an element m is encrypted by the session key x.

The detail descriptions of the messages in figure 3.2 are summarised in table 3.2.

3.2 Intended Properties of CPV02 Protocol

The following section is recalled from CPV02 [23], the thesis uses the terminologies that shown in that paper. In order to analyse the protocol, its properties have been clarified. These following properties have been modelled and verified. The protocol intends to achieve the following properties.

1. *Effectiveness.* The accessed computing platform is given neither the user's stored biometric code nor the user's submitted biometric data until the integrity of both the computing platform and biometric reader are checked by the smart card.

   The protocol intends to assure the user that his stored biometric code and biometric data which is read for user's verification are not disclosed to the corrupt machine. It is accomplished by the integrity check from the smart card. In this protocol, the smart card is possessed and trusted by the user. To secure the biometric information, the smart card verifies the integrity value of the biometric reader and the computing platform.

2. *Correctness.* The biometric reader is not given the user's submitted

biometric data until the user is convinced of the correctness of both the computing platform and biometric reader integrity checking.

This property assures the user the trustworthiness of the biometric reader and the computing platform before he places his biometric data on the reader. This protocol proposes an approach to confirm the user the integrity check of the platform and the reader. He will not present his biometric data unless he is satisfied with the integrity check.

3. *Privacy of Biometric Data.* An unauthorised entity that can listen to a message between the smart card and computing platform, or between the biometric reader and computing platform, cannot obtain either the user's stored biometric code or the user's submitted biometric data.

   In this protocol, the privacy property ensures that devices or components that are not involved in the protocol could not obtain the biometric data and biometric code.

The detail of how to interpret the protocol and these security properties, in order to verify and analyse, is given in the later section.

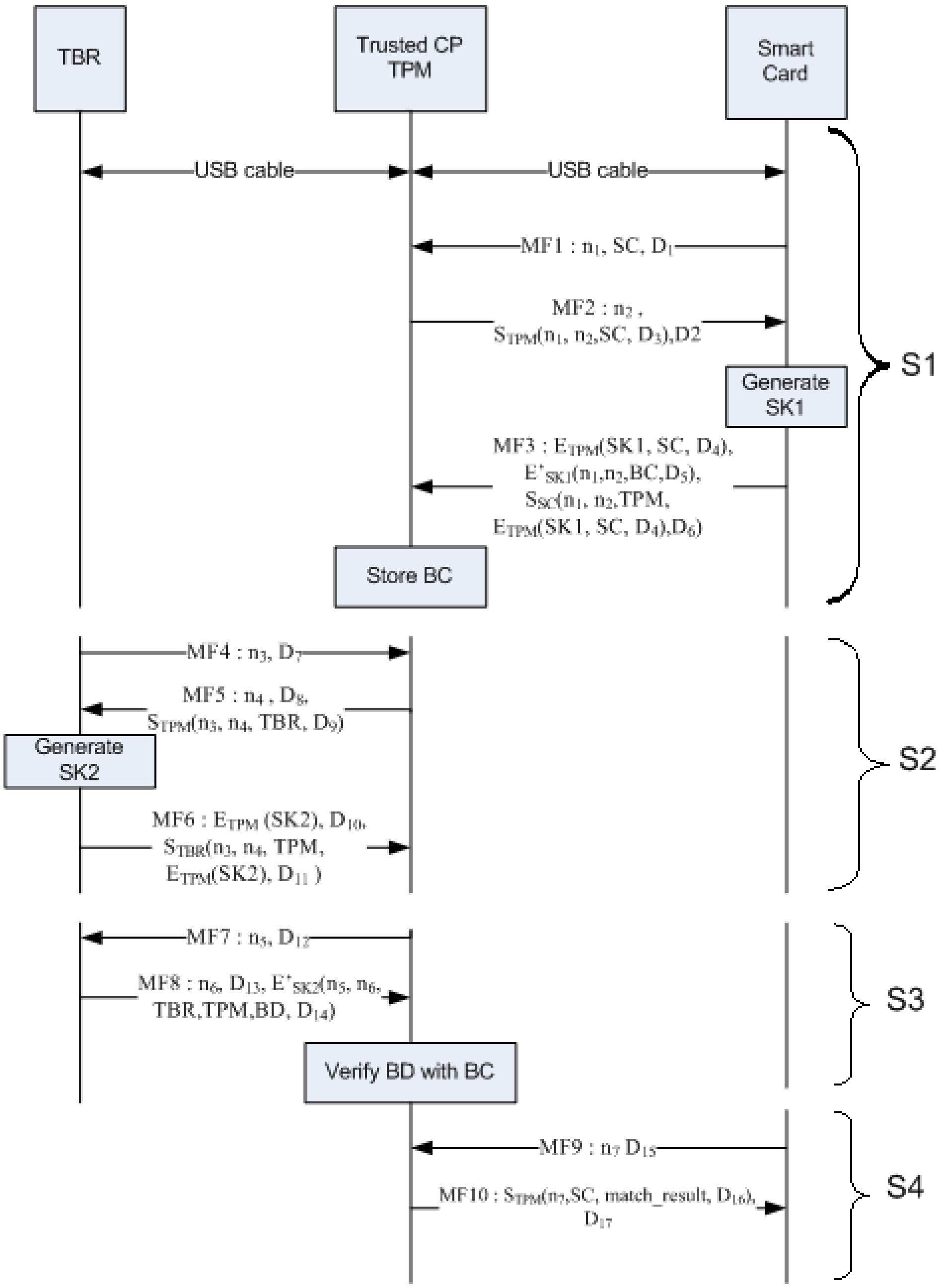
TBR
Trusted CP TPM
Smart Card
USB cable
USB cable
MF1 : n_1, SC, D_1
MF2 : n_2 , $S_{TPM}(n_1, n_2, SC, D_3)$,D2
Generate SK1
MF3 : $E_{TPM}(SK1, SC, D_4)$, $E'_{SK1}(n_1, n_2, BC, D_5)$, $S_{SC}(n_1, n_2, TPM, E_{TPM}(SK1, SC, D_4), D_6)$
Store BC
S1
MF4 : n_3, D_7
MF5 : n_4 , D_8, $S_{TPM}(n_3, n_4, TBR, D_9)$
Generate SK2
MF6 : E_{TPM} (SK2), D_{10}, $S_{TBR}(n_3, n_4, TPM, E_{TPM}(SK2), D_{11})$
S2
MF7 : n_5, D_{12}
MF8 : n_6, D_{13}, $E'_{SK2}(n_5, n_6, TBR, TPM, BD, D_{14})$
Verify BD with BC
S3
MF9 : n_7 D_{15}
MF10 : $S_{TPM}(n_7, SC, match\_result, D_{16})$, D_{17}
S4

Figure 3.2: Message Sequence Chart for CPV02 Protocol

Table 3.2: Summarisation of the Encrypted Messages in figure 3.2

Encrypted Message	Description
S_{TPM}(n1,n2,SC,D3)	nonce n1, nonce n2, the identity of the smart card and the integrity metric D3 are signed by the TPM's signature.
E_{TPM}(SK1,SC,D4)	the session key SK1, the identity of the smart card and the integrity metric D4 are encrypted by the public key of the TPM.
E'_{SK1}(n1,n2,BC,D5)	nonce n1, nonce n2, the biometric code and the integrity metric D5 are encrypted by the session key SK1.
S_{SC}(n1,n2,TPM, E_{TPM}(SK1,SC,D4), D6))	the session SK1, the identity of the smart card, and the integrity metric D4 are encrypted by the TPM's public key. Those encrypted package, nonce n1, nonce n2, the identity of the TPM and the integrity metric D6 are signed with a private key of the smart card.
S_{TPM}(n3,n4,TBR,D9)	nonce n3, nonce n4, the identity of the biometric reader and the integrity metric D9 are signed with a private key of the TPM.
E_{TPM}(SK2)	the session key SK2 is encrypted by the TPM's public key.
S_{TBR}(n3,n4, TPM,E_{TPM}(SK2),D11)	nonce n3, nonce n4, the identity of the TPM, the encrypted session key SK2 and the integrity metric D11 are signed with a private key of the biometric reader.
E'_{SK2}(n5,n6,TBR, TPM,BD,D14)	nonce n5, nonce n6, the identity of the biometric reader, the identity of the TPM, the biometric data and the integrity metric D14 are encrypted by the session SK2.
S_{TPM}(n7,SC, matchResult,D16)	nonce n7, the identity of the smart card, the matching result, the integrity metric D16 are signed with the private key of the TPM.

3.3 Problems Encountered

To verify the three protocol properties presented in section 3.2, we need to gain a detailed understanding of how the protocol works and the sequence of messages.

If a naive verifier were to interpret the CPV02 protocol as it is presented in [23] (page 7-9), it would identify an attack.

The sub-protocols are presented in a sequence order, and since nothing is said about the order in which they should be run, the reader can assume they are run in the order presented. We performed the verification on that assumption. The result of the verification shows that one of the properties does not hold: the biometric data is released before the TPM is checked.

3.3.1 ProVerif Model of the Naive Interpretation

The ProVerif model of this interpretation is consisted of four processes (excluding the main process): TPM, TBR, SC and ProcessK. Processes TPM, TBR and SC perform the operations of the TPM, TBR and SC respectively (functions of the components are described in section 3.1). ProcessK distributes the verification key certificates to the three processes TPM, TBR and SC. The main process generates private keys for each component and distributes them via private channels, running these processes concurrently. The ProVerif interpretations of processes are shown as follows:

Signature and Equational Theory

The signature and equational theory of this verification is represented in figure 3.3. The protocol uses both symmetric and public key cryptography. Symmetric cryptography are modelled as *sdec()* and *senc()*. The *sdec()* is used for deciphering a message that is encrypted by *senc()*. Asymmetric cryptography is modelled in ProVerif as *dec()* and *enc()*. The signature of a component is created by function *sign()* whereas a signed message is extracted by *checksign()* function.

Main Process

The main process generates secret keys for each identity, creates a public key for the CA, and distributes them. We modelled the protocol on the basis that there are many components and that they run concurrently. Therefore, the main process replicates these components and runs them concurrently. The main process is represented in figure 3.4

TPM Process

The process TPM starts by receiving the certificate via a private channel (as represented in figure 3.5. This certificate is used to certify the TPM's private key. The TPM checks that the component it is communicating with is the SC. The TPM waits for the session key created by the SC to share between them for decryption of the biometric code. When the biometric code is decrypted and stored, the ProVerif model executes event *tcpgetBC()*. Later on, the biometric reader is verified and the event *tbrchecked()* is executed. Event commands are executed during the process for property analysis. The second session key is created by the biometric reader and used for decryption of the biometric data, which is later received. Once the biometric data is admitted, event *tcpgetBD()* is executed. The matching process is performed, the matching result is generated, and sent out to the SC.

SC Process

The SC process represents the smart card's functionality in the protocol. The SC checks whether it is communicating with the desired component: the TPM. Once the signature of the received message is verified, the integrity of the TPM is examined. If the integrity value is agreed, event *tcpChecked()* is executed, the session key (SK1) for encryption of the biometric code is generated. The session key SK1 and the biometric code are sent out in an encrypted format. The SC then waits for the matching result. The ProVeif model of this process is shown in figure 3.6

TBR Process

The TBR process creates a session key (SK2) for encrypting the biometric data. This key is sent out to the TPM; it is encrypted using the public key of the TPM. The origin of the message is guaranteed by the TBR's signature. The user places his biometric data on the biometric reader causing the BD to be generated. This data is sent out in encrypted format. Figure 3.7 shows the ProVerif model of the TBR process

Key Distribution Process

The processK is the key distribution process which intends to deliver its own identity and key to each component via a private channel. The ProVerif representation of this process is shown in figure 3.8. This process aims to distribute the key securely. Its own identity and key is signed by the CA; therefore, each component has confidence in the correctness of the information.

3.3.2 Analysis Result from the Naive Interpretation

The result of the verification shows that the TCP is sent the BC before the TBR is checked. This breaks one of the intended properties of the protocol: *effectiveness*.

```
query ev:  tcpgetBC() ==>ev:  tcpChecked() & ev :  tbrChecked().
```

3.4 The Clarified CPV02 Protocol

Email discussion with one of the authors of [23], Liqun Chen, has given us further vital information about CPV02. We have learnt that the four sub-protocols can run at any time and in any order. Moreover, the result from one sub-protocol may affect the other sub-protocols. For example, sub-protocol (S1) cannot be run successfully without also running sub-protocol (S2). These facts cannot be easily extracted from the paper without the discussion and they are important in order to successfully verify the protocol.

Let us consider the message sequence chart of CPV02 in figure 3.2. The protocol consists of four sub-protocols (S1), (S2), (S3), and (S4) which can run in any order and at any time. In (S1), the encrypted BC is sent from the SC to the TPM. In (S2), a session key is created for use between the TPM and the TBR when the BD is encrypted. In (S3), the encrypted BD is sent from the TBR to the TPM. In (S4), a matching result on the BC and BD is sent from the TPM to the SC.

The detailed ProVerif model for verifying the properties according to the clarified protocol is presented in section 3.5.

3.5 Modelling the Clarified CPV02 in ProVerif

Now we model the CPV02 protocol based on the derived message sequence chart (shown in figure 3.2) from clarification and the following assumptions:

1. All the components, TPM, SC and TBR, hold the public key of certificate authority.

2. The integrity metric measurements have been made and are stored in the tamper-resistant storage. Therefore we model it, as it is a stored secret value, and verify its correctness with the challenger's stored value.

The ProVerif code consists of signature and equational theory, a main process, a process for certificate distribution, S1 process, S2 process, S3 process, and S4 process. A detailed description of each process will be given in a later section.

3.5.1 Signature and Equational Theory

Our ProVerif model involves public key and host functions. We model cryptographic function as *enc* and *dec*. Similarly, the symmetric cryptography is modelled as *senc* and *sdec*. In order to introduce digital signature, function *sign* is added and function *checksign* is used to verify the origin of the messages.

The public key cryptography is represented as `equation dec(enc(x,pk(y)),y) = x.` This equation enables ProVerif to decipher a message from the encrypted message if the public key is known. By providing `equation dec(enc(x,pk(y)),y) = x.`, ProVerif extracts encrypted messages from symmetric cryptography in a similar way. In the interests of verifying the origin of the messages; the `checksign` equation is introduced into our model. Figure 3.9 shows the functions and equations used in verification.

3.5.2 Main Process

Figure 3.10 shows the ProVerif model of verification of the main process; the public keys, private keys, and the identities of each component are created and distributed in the public channel. Moreover, the components are replicated and they are run at any time and in any order.

3.5.3 Certificate Distribution

This process (in figure 3.11) is intended to distribute the certificates of verification keys for the integrity checking process and distribute them through the private channel to guarantee that each identity will obtain them correctly.

3.5.4 (S1) Sending the Encrypted Biometric Code

This sub-protocol intends to transfer the biometric code which is stored in the smart card to the TPM. The smart card firstly requests the integrity metric from the TPM so that it ensures the security level of the TPM. The TPM returns the signed integrity metric. If the smart card is satisfied with the provided integrity metric, it then creates a session key and is sent to the TPM in encrypted format as well as the biometric code, which is encrypted with this key. The detailed model can be described in figure 3.12.

The sub-protocol S1 includes two processes: TPM1 and SC1. The mutual authentication between the TPM and the SC is performed before the encrypted BC is transmitted. Firstly, the TPM and the SC obtain their certificates. The TPM generates a fresh random nonce. Then it sends its integrity metric with this nonce to the SC. The SC checks the certificate it receives from the TPM and retrieves the public key of the TPM. The SC verifies the validity of messages and generates a session key and then sends the BC, encrypted by the key, to the TPM. The TPM verifies the accuracy of the received message, decrypts it, and stores the BC in its secure storage. Moreover, from email discussions, we have learnt that (S1) cannot run successfully before (S2) has run. So we add state checking to check that (S2) has run.

3.5.5 (S2) Creating a Session Key for Encrypting the User's Submitted Biometric Data

This sub-protocol generates another session key which will be used between the biometric reader and the TPM. The TPM checks whether the biometric reader meets the required integrity metric. For this protocol, it is assumed that the TPM has extended capability to verify that the biometric reader is trusted. The session key is generated by the biometric reader for encrypting biometric data later.

The sub-protocol S2 represents mutual authentication between the TPM and the TBR. It also creates a session key for sharing between the TPM and the TBR. This sub-protocol runs when a TBR has been introduced to the system.

This sub-protocol includes the two processes TPM2 and TBR2. The certificates are obtained via the private channels. Note that the TPM has already obtained this certificate in the previous sub-protocol. TPM1 and TPM2 are indeed the same trusted platform modules but they are run in different sub-protocols and therefore require distinct names. So we model TPM2 to receive the certificate again but the certificate it receives is the same certificate as that received by TPM1.

The TBR has to authenticate itself to the TPM using an integrity checking mechanism. It creates a fresh random number and sends it with its integrity metric. If the TPM is satisfied with the checking result, it will send its certificate along with the authentication message to the TBR. The TBR retrieves the public key of the TPM. It then checks the correctness of the message. If it is valid, the TBR will create a session key SK2 for the encryption and decryption of the BD.

While the processes TPM1, TPM2, TPM3, and TPM4 are on the same trusted platform module, in order to fit the CPV02 protocol they need to run as separate sub-protocols. This fact also applies to TBR2 and TBR3. All variables created or received in one TPM process should be known to others. Hence, in the process TPM2, two private channels are set up. One is used for acknowledging that S2 has run and the other is used for transferring the session key SK2 from the process TPM2 to the process TPM3. Similarly, a private channel is set up in process TBR2 to transmit the session key from the process TBR2 to the process TBR3. The ProVerif model of this sub-protocol can be found in figure 3.13.

3.5.6 (S3) Sending Encrypted User's Submitted Biometric Data From the Biometric Reader to the Trusted Platform Module

Once the session key is generated and sent out to the TPM, the TPM requests the biometric data from the biometric reader. The reader sends this data in encrypted format to the TPM.

The processes TPM3 and TBR3 run in sub-protocol (S3). From figure 3.14, firstly, the TPM obtains the session key SK2 via the private channel. The TBR also obtains the identity of the TPM and the session key via the private channels from TBR2.

The TPM generates a fresh random nonce and sends it to the TBR. Again, from email discussions about the sequence of the processes, (S3) cannot run successfully before (S1) has run. The TBR verifies the message and sends back the BD encrypted by the session key created at the previous stage. The TPM verifies the received message and decrypts it to retrieve the BD. In order to check protocol properties later, after the BD is received, an *event tcpgetBD()* is launched.

3.5.7 (S4) Sending a Matching Result

The last sub-protocol (S4) represents the transfer of a matching result on the BC and BD from the TPM to the SC.

After the biometric matching process is performed in the TPM, the smart card requests the result from the TPM. The response is signed by the TPM to guarantee its origin. This sub-protocol includes process TPM4 and process SC4. We model it (figure 3.15) to check the correctness of the messages received.

The TPM acquires its certificate via the private channel. The SC creates a fresh random number and sends it with a request. The TPM verifies the message. It then signs the match result message and sends it to the SC. We model a match result as a fresh value since we are not concerned with the mechanism by which the TPM carries out the matching process. The SC will check the signature and the correctness of the message. If it is correct, the SC may release the user's credentials to the TPM. We do not model how the SC releases these credentials since it goes beyond the definition of the protocol.

3.6 Analysis

As described earlier, if a naive interpretation of the protocol is applied, an attack is found. After the clarification of the protocol is introduced, we intend to analyse the properties of the protocol to see if the result of the verification is different.

We have analysed the three properties of CPV02, *effectiveness*, *correctness* and *privacy of biometric data*, using ProVerif. All three properties of the protocol are satisfied.

Using ProVerif as a verification tool means we can model a Dolev-Yao style attacker that can compose and decompose messages (provided it has relevant cryptographic keys), and has full control over messages that pass over public interfaces and networks.

In the case of the CPV02 protocol, the USB cables are considered as part of the public network, since an attacker can interfere with them. The smart card interfaces are also considered public. An example of an attack to the smart card could be illustrated as a prototype device presented in [10] showing that the device can listen to the signal between smart card and smart card reader. The device does not capture the credential information but relay the information among the legitimate user, an intruder and the system in order to gain an intruder's desire without user's consent. Detailed discussion of this can be found in section 2.5.3.

3.6.1 Effectiveness

The TCP will not be given to either the BC or the BD unless the integrity of the TPM and TBR has been checked by the SC.

According to the protocol, the BC is transferred from the SC to the platform, and the BD is read from the TBR and sent to the platform; then the two are compared. To protect the BD from a malicious attacker, the device holding this data has to be convinced that the destination to which it will transfer the data can be trusted before the transmission is carried out. This is done by means of integrity checks.

To analyse this property, we use the *event* and *query* commands. These two commands are used to check the correctness of sequences of events. While the *event* command is used for launching an event when a certain action is executed, the *query* command is used to prompt ProVerif to verify the correctness of the sequence of events that we specify. If the sequence is not correct, an attack is identified.

In order to verify this property in ProVerif, we encode the integrity check which ensures that the SC is satisfied with the integrity metric of the TCP and the TBR before the trusted platform module receives the user's stored biometric code and user's submitted biometric data.

The event *tcpChecked()* is inserted after the SC has checked the integrity of the TCP via the TPM, and the event *tcpgetBC()* is inserted after the TCP has received the BC.

Similarly, to verify that the integrity metric of the TBR is checked by the SC before the BD is transferred, an event *tbrChecked()* is launched after the SC has checked the integrity metric of the TBR.

It should be noted that there is no direct communication between the TBR and the SC, so the TPM is responsible for checking the integrity metric of the

TBR on behalf of the SC. To model this situation, we code it in such a way that if the TPM is satisfied with the integrity metric of the TBR, an event *tbrChecked()* is triggered. The TBR then sends the encrypted BD to the TPM. The TPM verifies the message, stores the BD, and then an event *tcpgetBD()* is inserted.

We need to check that these events are executed in the correct order, i.e. that the TPM's integrity metric and the TBR's integrity metric have been examined before the TPM receives the BD. This should be the case even in the presence of an attacker that can control the order of the subprotocols and the messages on the network. This check is implemented using ProVerif's *query* command:

```
query ev:  tcpgetBD() ⇒ ev:  tcpChecked() & ev :  tbrChecked().
query ev:  tcpgetBC() ⇒ ev:  tcpChecked() & ev :  tbrChecked().
```

3.6.2 Correctness

The TBR is not given the BD until the user is satisfied with the integrity checks on both the TCP and TBR.

This property aims to protect the BD from being read by a malicious biometric reader, the user places her biometric data only on the biometric reader that she trusts. This property is important because if the BD is stolen or accidentally disclosed, it cannot be altered, replaced or regenerated.

To verify this property, we check that the biometric reader (TBR) receives the BD after the integrity metric of the TCP and the integrity metric of the TBR have been checked.

To achieve this, we launch an event *tbrgetBD()* after the BD is created in the process TBR3.The event would not be triggered without satisfactory integrity checking. To check the correct order of events, we use the query command:

```
query ev: tbrgetBD() ⇒ ev: tcpChecked() & ev : tbrChecked().
```

3.6.3 Privacy of Biometric Data

An unauthorised entity that can listen to a message between the SC and TCP, or between the TBR and TCP, cannot obtain either the BC or the BD.

As we remarked earlier, the secrecy of biometric data cannot be relied upon. The security of a protocol should not depend on the secrecy of biometric data. Indeed, this protocol does not depend on it, since it uses a trusted biometric reader to guard against disclosure. Nevertheless, it is good practice to prevent widespread dissemination, and this property verifies that the protocol does not give an attacker easy access to that data.

To model this property we use the *query* command to ask ProVerif whether an attacker can access the BC or the BD. The commands for this verification are

```
query attacker : BC.
query attacker : BD.
```

Using these commands to check whether an attacker can gain the specified arguments, ProVerif will exhaustively check whether there is any way that an attacker could obtain the information, BD and BC, that the protocol wishes to keep private. If an attacker can obtain the data, then a potential attack has been identified.

3.7 Chapter Summary

We have presented a specification of the CPV02 biometric authentication protocol, obtained after clarifying details of the protocol through email discussion with one of the authors. We modelled the clarified protocol using the applied pi calculus and the ProVerif verification tool. We have encoded three intended properties of the protocol, namely *effectiveness*, *correctness* and *privacy of biometric data*. The positive results from the verification show that the properties of the protocol hold.

The protocol is successfully verified against the properties. Without this clarification, verification of one of the properties fails.

The CPV02 protocol uses a trusted computing platform and involves integrity checking. The trusted computing platform module is an essential part of the protocol in order to guarantee that the components that are involved in biometric authentication data cannot be tampered with by an intruder. Similar to other classical protocols, nonces are used for checking the freshness of messages received and encryption and decryption are also used for the secrecy of message content.

In the next chapter, we will select other protocols with different properties and verify that they hold in a similar way. We would also like to investigate biometric authentication protocol which can be used for unsupervised remote authentication, such as in on-line banking.

```
(* SIGNATURE *)
fun dec/2.
fun enc/2.
fun sdec/2.
fun senc/2.
fun pk/1.
fun checksign/2.
fun sign/2.
(* EQUATION *)
equation dec(enc(x,pk(y)),y) = x.
equation sdec(senc(x,k),k) = x.
equation checksign(sign(x,y),pk(y)) = x.
```

Figure 3.3: Signature and Equational Theory for the Naive Interpretation

```
process
(* Create secret keys *)
new skCA;
new skSC;
new skTPM;
new skTBR;
(* Create public key of certificate authority *)
let pkCA = pk(skCA) in
out(ch,pkCA);
(* Create identity of components *) let hostSC = host(skSC) in
let hostTPM = host(skTPM) in
let hostTBR = host(skTBR) in
out(ch,hostSC);
out(ch,hostTPM);
out(ch,hostTBR);
!(TPM) | !(SC) | !(TBR) | !(processK)
```

Figure 3.4: Main Process of the Naive Interpretation

```
let TPM =
in(privChCertTPM1,D2); (*The TPM stores the certificate for verification *)
(*The smart card authenticates the TPM. If the TPM is valid, the smart card
will release the biometric code to be stored in the TPM storage *)
in(ch,(nx1,hostSCx,=D1));
new n2;
out(ch,(n2,sign((nx1,n2,hostSCx,D3),skTPM),D2));
in(ch,(m2,m3,m4));
let(SK1Received,=hostSCx,=D4) = dec(m2,skTPM) in
let(=nx1,=n2,BCReceived,Dx5) = sdec(m3,SK1Received) in
let(=hostSCx,pkSCx) = checksign(Dx5,pkCA) in
let(=nx1,=n2,=hostTPM,m5,=D6) = checksign(m4,pkSCx) in
let(=SK1Received,=hostSCx,=D4) = dec(m5,skTPM) in
event tcpgetBC(); (* Create the event to specify that the TCP has received biometric code *)
let D8 = D2 in (* The integrity metric of the biometric reader is verified *)
in(ch,(nx3,Dx7)); let(hostTBRx,pkTBRx) = checksign(Dx7,pkCA) in
if hostTBRx = hostTBR then
event tbrChecked();(* Create the event to specify that the biometric reader is valid *)
(*The TPM receives the encrypted biometric data which is read from the biometric reader *)
new n4; out(ch,(n4,D8,sign((nx3,n4,hostTBRx,D9),skTPM)));
in(ch,(m7,=D10,m8));
let(SK2) = dec(m7,skTPM) in
let(=nx3,=n4,=hostTPM,m9,=D11) = checksign(m8,pkTBRx) in
let(=SK2) = dec(m9,skTPM) in
let D12 = D2 in
new n5;
out(ch,(n5,D12));
in(ch,(nx6,Dx13,m10));
let(=hostTBRx,=pkTBRx) = checksign(Dx13,pkCA) in
let(=n5,=nx6,=hostTBRx,=hostTPM,BDReceived,=D14) = sdec(m10,SK2) in
event tcpgetBD();
(* After the biometric data has been received, the biometric data verification is performed *)
let D17 = D2 in
in(ch,(nx7,Dx15));
let(=hostSCx,=pkSCx) = checksign(Dx15,pkCA) in
new matchResult;
out(ch,(sign((nx7,hostSCx,matchResult,D16),skTPM),D17)).
```

Figure 3.5: TPM Process of the Naive Interpretation

```
let SC =
in(privChCertSC1,D5); (*The smart card stores the certificate for verification *)

(* The smart card checks the validity of the TCP. The smart card releases
the stored biometric code if it satisfies with the TCPs integrity metric.
The biometric code, encrypted by the shared session key, is transmitted to the TPM *)
new n1;
out(ch,(n1,hostSC,D1));
in(ch,(nx2,m1,Dx2));
let(hostTPMx,pkTPMx) = checksign(Dx2,pkCA) in
let(=n1,=nx2,=hostSC,imtpmReceived) = checksign(m1,pkTPMx) in
if imtpmReceived <> D3 then 0
else(
event tcpChecked(); (* Create the event to specify that TCP is verified. *)
new SK1;
new BC;
out(ch,(enc((SK1,hostSC,D4),pkTPMx),senc((n1,nx2,BC,D5),SK1),
sign((n1,nx2,hostTPMx,enc((SK1,hostSC,D4),pkTPMx),D6),skSC)));
let D15 = D5 in
new n7;
out(ch,(n7,D15));
in(ch,(m12,Dx17));
let(=hostTPMx,=pkTPMx) = checksign(Dx17,pkCA) in
let(=n7,=hostSC,matchResultReceived,=D16) = checksign(m12,pkTPMx) in
).
```

Figure 3.6: SC Process of the Naive Interpretation

```
let TBR =
in(privChCertTBR2,D7); (* The biometric reader stores the certificate for verification *)
(* After the biometric reader has been verified, the biometric data is read
and encrypted by the newly generated shared session key. The encrypted data is sent to the TPM *)
new n3;
out(ch,(n3,D7));
in(ch,(nx4,Dx8,m11));
let(hostTPMz,pkTPMz) = checksign(Dx8,pkCA) in
let(=n3,=nx4,=hostTBR,Dx9) = checksign(m11,pkTPMz) in
if Dx9 <> D9 then 0
else(
new SK2;
out(ch,(enc((SK2),pkTPMz),D10,
sign((n3,nx4,hostTPMz,enc((SK2),pkTPMz),D11),skTBR)));
let D13 = D7 in
in(ch,(nx5,Dx12));
let(=hostTPMz,=pkTPMz) = checksign(Dx12,pkCA) in
new n6;
new BD;
out(ch,(n6,D13,senc((nx5,n6,hostTBR,hostTPMz,BD,D14),SK2)))
).
```

Figure 3.7: TBR Process of the Naive Interpretation

```
let processK =
(* The secret keys of the TPM, smart card and the biometric reader are
generated and distributed to each component safely via private channels *)
out(privChCertTPM1,sign((host(skTPM),pk(skTPM)),skCA))|
out(privChCertSC1,sign((host(skSC),pk(skSC)),skCA))|
out(privChCertTBR2,sign((host(skTBR),pk(skTBR)),skCA)).
```

Figure 3.8: Key Distribution Process of the Naive Interpretation

```
(* SIGNATURE *)
fun dec/2.
fun enc/2.
fun sdec/2.
fun senc/2.
fun pk/1.
fun checksign/2.
fun sign/2.
(* EQUATION *)
equation dec(enc(x,pk(y)),y) = x.
equation sdec(senc(x,k),k) = x.
equation checksign(sign(x,y),pk(y)) = x.
```

Figure 3.9: Signature and Equational Theory for the Analysis of [23]

```
Process
(* Create secret keys *)
new skCA;
new skSC;
new skTPM;
new skTBR;
(* Create public key *)
let pkCA = pk(skCA) in
out(ch,pkCA);
(* Identify the components identities *)
let hostSC = host(skSC) in
let hostTPM = host(skTPM) in
let hostTBR = host(SKTBR) in
out(ch,hostSC);
out(ch,hostTPM);
out(ch,hostTBR);
!(TPM1) | !(TPM2) | !(TPM3) | !(TPM4)|
!(SC1) | !(SC4) |
!(TBR2) | !(TBR3) |
!(processK)
```

Figure 3.10: Main Process for the Analysis of [23]

```
let processK =
(* The private key and the identity of each component are signed by the CA
and distributed via private channel *)
out(privChCertTPM1,sign((host(skTPM),pk(skTPM)),skCA))|
out(privChCertTPM2,sign((host(skTPM),pk(skTPM)),skCA))|
out(privChCertTPM3,sign((host(skTPM),pk(skTPM)),skCA))|
out(privChCertTPM4,sign((host(skTPM),pk(skTPM)),skCA))|
out(privChCertSC1,sign((host(skSC),pk(skSC)),skCA))|
out(privChCertSC4,sign((host(skSC),pk(skSC)),skCA))|
out(privChCertTBR2,sign((host(skTBR),pk(skTBR)),skCA)).
```

Figure 3.11: Certificate Distribution for the Analysis of [23]

```
let TPM1 =
in(pState2,P2);
if P2 <> success then 0
else (
in(privChCertTPM1,D2); (* The certificate for verification is stored in the TPM*)
in(ch1,(nx1,hostSCx,=D1));
new n2;
out(ch2,(n2,sign((nx1,n2,hostSCx,D3),skTPM),D2));
in(ch3,(m2,m3,m4));
let(SK1Received,=hostSCx,=D4) = dec(m2,skTPM) in
let(=nx1,=n2,BCReceived,Dx5) = sdec(m3,SK1Received) in
let(=hostSCx,pkSCx) = checksign(Dx5,pkCA) in
let(=nx1,=n2,=hostTPM,m5,=D6) = checksign(m4,pkSCx) in
let(=SK1Received,=hostSCx,=D4) = dec(m5,skTPM) in
event tpmgetBC()
).
```

```
let SC1 =
in(privChCertSC1,D5); (* The certificate for verification is stored in the smart card *)
(* The smart card authenticates the TPM if it is trusted to transfer the biometric code to *)
new n1;
out(ch1,(n1,hostSC,D1));
in(ch2,(nx2,m1,Dx2));
let(hostTPMx,pkTPMx) = checksign(Dx2,pkCA) in
let(=n1,=nx2,=hostSC,imtpmReceived) = checksign(m1,pkTPMx) in
if imtpmReceived <> D3 then 0
else(
(* If the smart card satisfies with the validity of the TPM, the smart card
will release the biometric code to the TPM. The biometric code is transferred
in encrypted format by using shared session key *)
event tpmChecked();
new SK1;
new BC;
out(ch3,(enc((SK1,hostSC,D4),pkTPMx),senc((n1,nx2,BC,D5),SK1),
sign((n1,nx2,hostTPMx,enc((SK1,hostSC,D4),pkTPMx),D6),skSC)));
out(pState1,success)
).
```

Figure 3.12: (S1) Process for the Analysis of [23]

```
let TPM2 =
in(privChCertTPM2,D8); (* The certificate for verification is stored in the TPM *)
(* The TPM verifies the validity of the biometric reader. Then, a nonce is
generated in order to use later to verify the session key which is received from the biometric reader *)
in(ch4,(nx3,Dx7));
let(hostTBRx,pkTBRx) = checksign(Dx7,pkCA) in
if hostTBRx = hostTBR then
event tbrChecked();
new n4;
out(ch5,(n4,D8,sign((nx3,n4,hostTBRx,D9),skTPM)));
in(ch6,(m7,=D10,m8));
let(SK2) = dec(m7,skTPM) in
let(=nx3,=n4,=hostTPM,m9,=D11) = checksign(m8,pkTBRx) in
let(=SK2) = dec(m9,skTPM) in
out(pState2,success);
out(privSK2TPM2,SK2).
```

```
let TBR2 =
in(privChCertTBR2,D7); (* The certificate for verification is stored in the biometric reader *)
(* The biometric reader verifies itself with the TPM and creates a session
key for biometric data encryption. This key is sent to the TPM *)
new n3;
out(ch4,(n3,D7));
in(ch5,(nx4,Dx8,m11));
let(hostTPMz,pkTPMz) = checksign(Dx8,pkCA) in
let(=n3,=nx4,=hostTBR,Dx9) = checksign(m11,pkTPMz) in
if Dx9 <> D9 then 0
else (
new SK2;
out(ch6,(enc((SK2),pkTPMz),D10,sign((n3,nx4,hostTPMz,enc((SK2),pkTPMz),
D11),skTBR)));
out(privSK2,SK2)
).
```

Figure 3.13: (S2) Process for the Analysis of [23]

```
let TPM3 =
in(privChCertTPM3,D12); (* The certificate for verification is stored in the TPM *)
in(privSK2TPM2,SK2TPM2);
(* The TPM generates a nonce and waits for the receipt of the encrypted
biometric data which is released from the biometric reader *)
new n5;
out(ch7,(n5,D12));
in(ch8,(nx6,Dx13,m10));
let(hostTBRxx,pkTBRxx) = checksign(Dx13,pkCA) in
let(=n5,=nx6,=hostTBRxx,=hostTPM,BDReceived,=D14) =
sdec(m10,SK2TPM2) in
event tpmgetBD().
```

```
let TBR3 =
in(pState1,P1);
if P1 <> success then 0
else (
in(privChCertTBR3,D13); (* The certificate of the verification is stored in the biometric reader *)
in(privSK2,SK2TBR3);
(After the integrity metric of the biometric reader is satisfied, the
biometric data is read and sent to the TPM *)
in(ch7,(nx5,Dx12));
let(hostTPMzz,pkTPMzz) = checksign(Dx12,pkCA) in
new n6;
new BD;
out(ch8,(n6,D13,senc((nx5,n6,hostTBR,hostTPMzz,BD,D14),SK2TBR3)))
).
```

Figure 3.14: (S3) Process for the Analysis of [23]

```
let TPM4 =
in(privChCertTPM4,D17); (* The certificate for verification is stored in the TPM *)
(* The TPM receives the request of the biometric verification result. The result is
signed and sent to the smart card *)
in(ch9,(nx7,Dx15));
let(hostSCxx,pkSCxx) = checksign(Dx15,pkCA) in
new matchResult;
out(ch10,(sign((nx7,hostSCxx,matchResult,D16),skTPM),D17)).
```

```
let SC4 =
in(privChCertSC4,D15); (* The certificate of verification is stored in the TPM *)
(* The smart card requests the biometric verification result from the TPM *)
new n7;
out(ch9,(n7,D15));
in(ch10,(m12,Dx17));
let(hostTPMxx,pkTPMxx) = checksign(Dx17,pkCA) in
let(=n7,=hostSC,matchResultReceived,=D16) = checksign(m12,pkTPMxx) in
```

Figure 3.15: (S4) Process for the Analysis of [23]

Chapter 4

Analysis of a Biometric Authentication Protocol for Signature Creation Application

This chapter presents a biometric authentication protocol which is created for a specific application, signature creation. This protocol is selected for the purpose of investigating and analysing a specific purpose protocol. Approaches and requirements for securing the biometric data are examined. This chapter illustrates the specifications of the protocol, the verification and analysis and discussion of attack to the protocol are presented. On top of that, the finding that the protocol extension is necessary in order to analyse the protocol is shown.

4.1 Description of the Protocol

This protocol is presented by Waldmann, Scheuerman and Eckert [7]. The protocol prevents the user's biometric data from bypassing a biometric reader and protects the data package using a cryptographic mechanism.

A signature creation application that stores the user's biometric code on a smart card is used here to illustrate this protocol. This application enables the user to sign a document using his private key. The user's private key is stored on the smart card. It will be released if the user is successfully verified by using his biometric data.

The physical setup of the system is shown in figure 4.1. The system consists of a PC and a terminal case. The PC contains a service application such as the signature creation application. Inside the terminal case are the security module card (SMC), tamper resistant terminal, and user card containing the user's credentials. In order to prevent fraud and interruption from an intruder, the fingerprint reader (including biometric feature extraction), secured viewer and smartcard interaction module (SIM) are embedded in the tamper resistant terminal.

Let us describe the biometric authentication process that takes place when the user wishes to sign a document using his signature. The user, Bob, uses his PC to open the signature creation application and he is shown a document via the secured viewer. If he agrees to sign it, he will present his biometric data - in this example, his fingerprint - to the sensor. The security protocol is performed via SIM in order to validate the user (detailed description is shown in the next section). If the user verification is successful, the user card will release Bob's signature to sign the document. The signing process is performed inside the tamper resistant terminal.

Figure 4.2 illustrates the processes involved in the security protocol. The three components of the system that perform the security functions are the SMC, the SIM and the user card. Table 4.1 summarises the notation, meaning and purpose of each component in this protocol.

The SMC is responsible for generating the session keys for encryption and decryption of biometric data. It is a plug-in card to give flexibility to the manufacturer of the service system. For example, the certificate of the service system can be changed easily, if necessary. The user card holds the user's

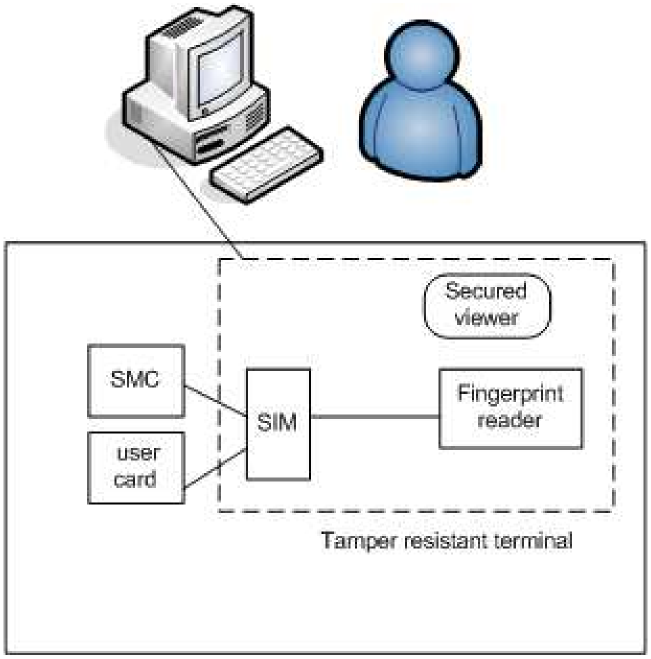

Figure 4.1: The Physical Setup of How Components are Connected

biometric code and other user credentials such as the user's signature if the service system is used for signature creation. The SMC and the user card cannot communicate directly and are outside the tamper resistant terminal so the SIM is responsible for the security protocol between the SMC and the user card.

Let us briefly describe how the protocol proceeds. The legitimate user, Bob, holds his user card, which stores his biometric code and private key. Before user authentication, the SMC and the user card perform mutual authentication, e.g. by using the Needham Schroeder Lowe protocol; if this succeeds, they will calculate the session keys `SK.CG` and `SK.CC`, and the initial value of the send sequence counter (`SSC`).

Apart from the new generated session keys, the SMC holds static keys, *`SK.CC` and *`SK.CG`, which are generated by the manufacturer. These keys are also installed in the SIM.

The `CC` which is included in the notation denotes the cryptographic check-

Table 4.1: Notations and Meanings [7]

Notation	Meaning/Purpose	Comments
SMC	Security Module Card	A plug-in card for key generation
SIM	Smartcard Interaction Module	Security Interface between SMC and User card
User Card	Store biometric code and perform matching process	Hold by user
SK.CC	Static key for cryptographic checksum	Use between SMC and SIM
SK.CG	Static key for biometric data encryption	Use between SMC and SIM
*SK.CC	Session key for cryptographic checksum	Use between SMC and User Card
*SK.CG	Session key for biometric data encryption	Use between SMC and User Card

sum for ensuring data integrity while the CG represents the cryptogram which is used for data encryption. Consider the following example that represents the message M, which is encrypted using key *SK.CG and then hashed using *SK.CC.

$$\{\mathtt{M}\}_{*SK.CG}\ ||\mathtt{H}_{*SK.CC}(\{\mathtt{M}\}_{*SK.CG})$$

The receiver of the above message could check the integrity of the received message by performing the hash function of the first argument and then comparing the result with the second argument. Moreover, the receiver could get the content of the message by performing message decryption using static key *SK.CG. The same idea applies to the message that uses the session key for encryption and hash function.

In order to sign a document using his electronic signature, Bob is shown the document via the secured viewer. The secured viewer is proposed in consideration of preventing an attacker that could interfere with the signal of the PC monitor. It is installed in the tamper resistant terminal so that an intruder could not interfere. If he agrees to sign it, he presents his fingerprint to the biometric reader that is situated in the tamper resistant terminal. To prevent replay of the presented biometric data, the SMC invents a fresh random nonce and sends it to the SIM to verify that the received message is fresh.

Before sending Bob's biometric data to the SMC, the SIM encrypts it with *SK.CG and also carries out the cryptographic checksum of encrypted user's biometric data using *SK.CC and the nonce.

After the SMC receives the message, it verifies its authenticity and validity. If this check is successful, it will send a reply ”OK” back to the SIM. The SIM then sends a sign command to user card.

The SMC calculates the cryptogram of the biometric data, and the cryptographic checksum of the cryptogram along with the user command by using the session keys (SK.CG and SK.CC) and sends this package to the SIM. On receipt, the SIM forwards this data package to the user card. The user card deciphers the package, checks the correctness, and verifies the received biometric data against the stored biometric code. Then the user card sends the result of the verification as well as the cryptographic checksum of the result back to the SMC via the SIM. The SMC verifies the correctness of the data it receives and the result of the user's verification. A positive result leads to the agreement of the signing process by the user card, the detail of which is beyond the scope of this protocol.

4.2 Extension of the Protocol for Signature Creation

It is necessary to extend the protocol, as described in the previous section, in order to completely verify the protocol and its properties. Here, we give some observations on the protocol and explain how the protocol should be completed in signature creation.

One of the purposes of the protocol is to enable the user to sign the document using the user's stored private key stored on the smart card. To verify

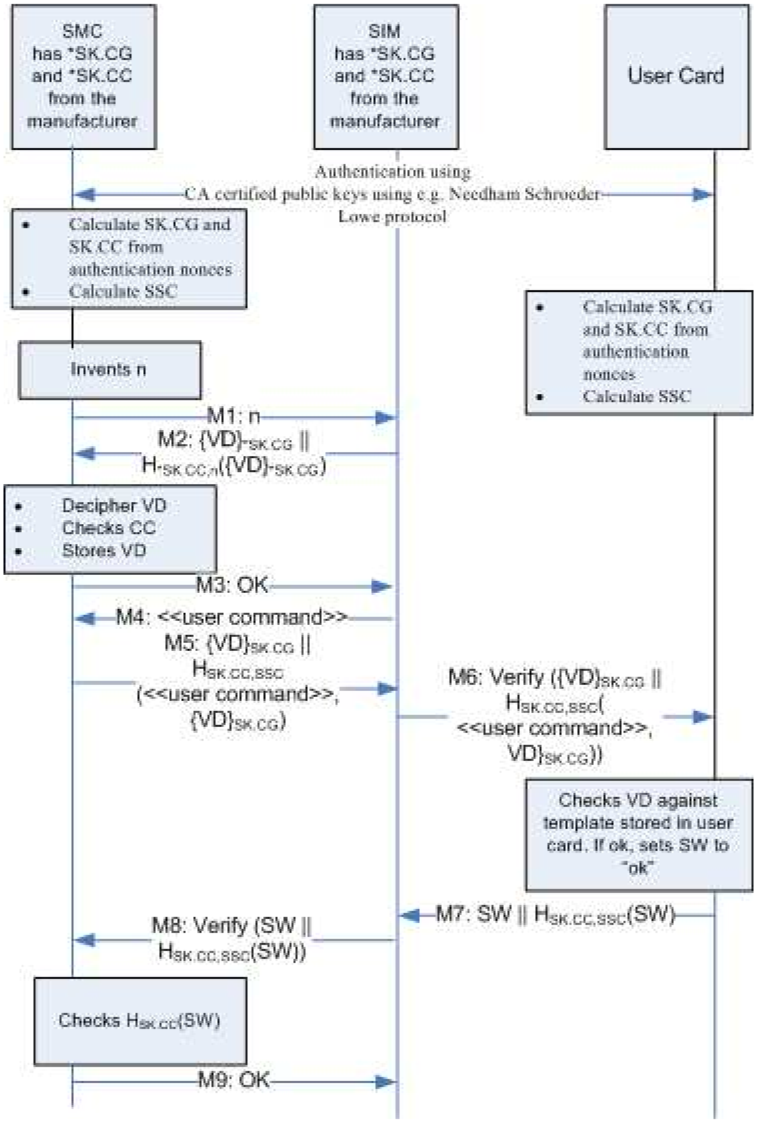
SMC has *SK.CG and *SK.CC from the manufacturer
SIM has *SK.CG and *SK.CC from the manufacturer
User Card
Authentication using CA certified public keys using e.g. Needham Schroeder Lowe protocol
Calculate SK.CG and SK.CC from authentication nonces
Calculate SSC
Calculate SK.CG and SK.CC from authentication nonces
Calculate SSC
Invents n
M1: n
M2: {VD}-SK.CG || H-SK.CC,n({VD}-SK.CG)
Decipher VD
Checks CC
Stores VD
M3: OK
M4: <<user command>>
M5: {VD}SK.CG || HSK.CC,SSC (<<user command>>, {VD}SK.CG)
M6: Verify ({VD}SK.CG || HSK.CC,SSC(<<user command>>, VD}SK.CG))
Checks VD against template stored in user card. If ok, sets SW to "ok"
M7: SW || HSK.CC,SSC(SW)
M8: Verify (SW || HSK.CC,SSC(SW))
Checks HSK.CC(SW)
M9: OK

Figure 4.2: The Message Sequence Chart of [7]

the protocol and guarantee correctness, the protocol has to be extended. After the user biometric authentication succeeds (more precisely after the message M9 has finished), in order to sign a document using the user's key, the SIM sends a hash value for the document to the SMC. The hash value of the document is encrypted with one of the static keys, *`SK.CG`. The SMC deciphers it and forwards the hash of the document, which is encrypted by the session key (shared by the SMC and the user card) `SK.CG` to the user card. The user

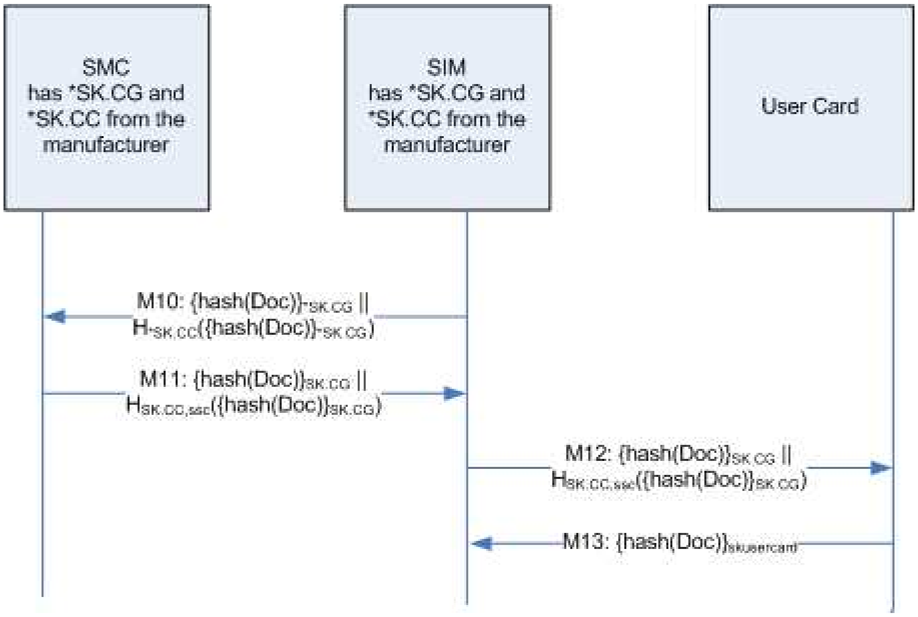

Figure 4.3: The Message Sequence Chart for Creating a Signature

card signs the hash value of the document and sends it back to the SIM. The document is signed only if the user is satisfied with the document he views from the terminal (via the secured viewer in the tamper resistant terminal). In accordance with signing a document, the rest of the protocol should be completed as shown in figure 4.3.

4.3 Capabilities of the Attacker

Again, a Dolev-Yao style attacker is used to model an attacker for the protocol. It can generate, mix, and replay messages transmitted in channels [8], even in cabling communication.

Biometric authentication uses a biometric reader in order to retrieve the user's biometric data. It is connected to the system via a USB cable. In addition, if a smart card is used to store the user's biometric code, a smart card reader is also connected to the system.

Although a smart card is a tamper resistant device in which the stored value cannot be modified without using the appropriate protocol, an attacker can still listen to the communication signal between smart card and reader. There is a prototype model that can be used as an example to describe this concept [10]. The detail description of this concept is described in section 2.5.3. The smart card itself does not have a display; it needs another device then, i.e. a smart card reader, to show any value to the user. Communication between the user and the smart card must take place via the reader. If it is modified by a corrupted merchant, information flow between the smart card and the card reader can be intercepted. So if the smart card is used for storing the biometric code for user verification, the attacker can listen to the messages and capture this data easily.

4.4 Verification of the Signature Creation Application

The protocol is verified using ProVerif. The ProVerif model consists of six parts: signature and equational theory, SIM process, SMC process, UserCard process, U process and Main process. ProVerif starts from Main process and run through other process to the end.

4.4.1 Signature and Equational Theory

In our model, ProVerif uses the signatures and equations shown in figure 4.4 for calculating and solving messages. The signed messages are extracted using the *checksign* equation. In order to retrieve the public key securely from the server, ProVerif uses the equation *getkey*. The function *getkey* will get the public key of the particular identity from the public key table which is stored in the server. In addition, *getkey* is a private function to prevent components other than those involving the system from using it in order to retrieve the key.

4.4.2 SMC Process

This process represents the operations and message transmission associated with the SMC. First, the SMC performs mutual authentication with the user card. It is not stated how this is done in [7]; we have used the Needham Schroeder Lowe protocol. If successful, it will calculate session keys (`SK.CG` and `SK.CC`) and `SSC` from the authentication nonces.

Figure 4.5 shows the ProVerif model of this process. The user's biometric data package is received and deciphered. Next, it encrypts and calculates the cryptographic checksum of the biometric data, and sends it to the SIM. If the user's authentication is successful, it will receive the verification result back from the user card, send the reply back to the SIM, and wait for the hash of the document to be sent back. After receiving the hash of the document, it will verify the validity of the document, encrypt, and calculate the cryptographic checksum of the hash of the document with the session keys `SK.CG` and `SK.CC` respectively.

4.4.3 SIM Process

In the real-life model, the user is presented with the document that he will sign using his key on the secured viewer. If he agrees to sign it, then he places his biometric data on the biometric reader which is installed in the SIM. Therefore, for ease of understanding, in the ProVerif model, the document that the user wants to sign is created within the SIM. The SIM receives a fresh random nonce and then sends the user's biometric data encrypted with *`SK.CG`, along with the cryptographic checksum created using *`SK.CC`, and the nonce, to the SMC. When the SIM receives the signal from the SMC that the user's biometric data is correct, it sends the user's command authorizing the signature as a reply.

The SIM carries out the security protocol between the SMC and the user card by receiving and forwarding messages between those two components.

After the user's authentication succeeds, the SIM generates the hash value of the document, encrypts it, and calculates its cryptographic checksum. It then sends these data to the SMC. The SMC is waiting to receive the document which is to be signed by the user card. Figure 4.6 represents the model of this process.

4.4.4 UserCard Process

First, the user card executes the mutual authentication with the SMC. Then, it calculates the session keys `SK.CG` and `SK.CC`, and `SSC`. Next, the user card awaits a package of the user's biometric data. It verifies the validity and authenticity of the received message. It decrypts the package and verifies the received biometric data against the stored biometric code. If they match, the verification result is set to be successful. In our ProVerif model (figure 4.7), they are always set to match so that we can verify the protocol until the end (the signing process of the document) without blocking through failure in biometric verification. The verification result is sent out along with the checksum of the result which is computed using `SK.CC`. Then, it acquires the hash of the document and signs it using the user's signature, which is stored in the user card.

4.4.5 U process

To demonstrate user interaction in the protocol, we model the process U shown in figure 4.8. The user receives a document and checks it. If he is satisfied with the contents, he will place his finger on the reader.

4.4.6 S process

In order to authenticate identities using the Needham Schroeder Lowe protocol, the server is modelled using process S, which is used for providing public key to the identity. The detail ProVerif model of the server process is shown in figure 4.9. The server process receives the request from the identity ”a” that it wants to communicate with identity ”b”. The server process retrieves the public key of the identity ”b” from the server's public key table. It then signs the package of the public key and the identity ”b” using its private key and outputs to the channel. The receiver of this package ensures that the public key it receives comes from the genuine server by checking the signature. The public key will be used later in the receiver process in order to perform the Needham Schroeder Lowe authentication which needs the public keys for decryption.

4.4.7 Main Process

In the main process, the static keys *`SK.CG` and *`SK.CC`, and the private keys of the SMC, the SIM and the user card are created. Private channels for the user's document and biometric data are set up. The public keys of each of the components are distributed on the public channels. In figure 4.10, there are many `SMC`, `SIM`, `user card`, and `U` processes in the system. The U processes represent Alice (an attacker) and Bob (the legitimate user) who input the documents and the biometric data.

4.5 Analysis of the Protocol

A signature application protocol is used as an example of using biometric authentication in order to verify the user who uses the smart card to sign a document that he is the correct user.

This thesis analyses two properties of this protocol: privacy of biometric data and intensional authentication. Analysis of this protocol considers an attack associated to a smart card; an intruder could interfere between the smart card and smart card reader to try to listen to the communication and capture user's biometric data [10]. Moreover, an intruder could play with messages to lead a legitimate user to sign her messages.

4.5.1 Privacy of Biometric Data

This property is used to verify that the protocol does not reveal the user's biometric data without permission. Even though we consider the biometric data to be public, it is good practice to keep it private so that no one else except the sender and the receiver know the content of messages. The protocol should not allow the data presented by user to be announced to others without his permission. Analysis of this property verifies whether an attacker can intercept the biometric data when it is sent from one component to another. In our model, the biometric data is represented as BobBD, the biometric data of legitimate user, Bob. The ProVerif implementation is:

```
query attacker :  BobBD
```

ProVerif responds to the *query* command by using a Dolev-Yao style attacker to attempt to compose or decompose messages and establish whether an attacker can reach the biometric data (BobBD).

4.5.2 Intensional Authentication

This property is used to verify that the document is signed only if the user's authentication is successful and that only the legitimate user signs the agreed document. The thesis analyzes this by checking whether an attacker can sign someone else's documents using the signature of the legitimate user. From our assumption, we check whether an intruder, Alice, can intercept messages to lead the legitimate user, Bob, sign her document. The ProVerif implementation is:

```
query attacker :  sign(AliceText,skBobUserCard)
```

ProVerif analyzes this *query* command by checking whether an attacker can sign AliceText (which is not the document that is shown to the legitimate user, Bob) using Bob's signature. We assume that the user's signature is the same as the private key of the user card that the user holds.

4.6 Chapter Summary

This chapter shows the specification and extension to the WSE04 [7]. This chapter presents verification and analysis of the two properties of the protocol: *privacy of biometric data* and *intensional authentication*. Without the extension, the protocol cannot be successfully verified against the intensional authentication property of the protocol. The verification shows that the protocol achieves its two intended properties.

The privacy property is verified to in the consideration that although we consider the biometric data to be public, we still need to verify that the protocol which uses this resource does not reveal it without the user's consent. The data should not be revealed to anyone who is neither the sender nor the intended receiver. The positive result of the verification illustrates that the presented biometric data remains private within the protocol and an attacker cannot acquire it.

The positive result of the intensional authentication property shows that the protocol guarantees that even if the presented biometric data is captured from the previous submitted data packet, it cannot lead the user card to sign a document that the user is not willing to sign.

The protocol WSE04 uses the session keys to secure the messages transmitted in the public channels, nonces are used to guarantee the freshness of the messages and the hash functions are employed for the messages validation.

```
(* SIGNAURE *)
private fun getkey/1.  (*key retrieval*)
fun sk/1.  (*session key*)
fun senc/2.  (*symmetric encryption*)
fun sdec/2.  (*symmetric decryption*)
fun enc/2.  (*encryption*)
fun dec/2.  (*decryption*)
fun sign/2.  (*signature *)
fun checksign/2.  (*recovering signature*)
fun pk/1.  (*public key*)
fun host/1.  (*host function*)
fun h/2.  (*hash function*)
fun g/2.  (*hash function *)
fun f/2.  (*hash function*)
fun hashDoc/1.  (*hash function for a document*)

(* EQUATION *)
equation dec(enc(x,pk(y)),y) = x.
equation sdec(senc(x,K),K) = x.
equation checksign(sign(x,y),pk(y)) = x.
equation getkey(host(x)) = x.
```

Figure 4.4: Signature and Equational Theory for the Analysis of [7]

```
let SMC =
(* The SMC performs mutual authentication with the user card;
the authentication method is not specified. In this case, we have used
Needham Schroeder Lowe Protocol in order to perform authentication *)
in(c,hostX);
let hostSMC = host(pkSMC) in
out(c,(hostSMC,hostX));
in(c,ms); let(pkX,=hostX) = checksign(ms,pkS) in
new Na; out(c,enc((Na,hostSMC),pkX));
in(c,m); let(=Na,NX2,=hostX) = dec(m,skSMC) in
out(c,enc(NX2,pkX));
let SKCG = h(Na,NX2) in
let SKCC = g(Na,NX2) in
let SSC = f(Na,NX2) in
(* After the authentication succeeds, the SMC receives the verification
package. The encrypted biometric data and its hash value is sent from the SIM to the SMC *)
new n;
out(c1,n);
in(c2,(m1,m2));
if h((sSKCC,n),m1) = m2 then
( let BDreceived = sdec(m1,sSKCG) in
out(c3,OK); in(c4,m13);
(* The decipher process for the encrypted data is carried out and the nonce
value is verified to check whether the data is fresh *)
let BDsenc = senc(BDreceived,SKCG) in
out(c5,(BDsenc,h((SKCC,SSC),(m13,BDsenc))));
in(c8,(m8,m9));
if h((SKCC,SSC),m8) = m9 then
if m8 = success then
out(c9,OK);
in(c10,(m16,m17));
if h(sSKCC,m16) = m17 then
( let M1 = sdec(m16,sSKCG) in
let M1senc = senc(M1,SKCG) in
out(c11,(M1senc,h((SKCC,SSC),M1senc)))
)).
```

Figure 4.5: SMC Process for the Analysis of [7]

```
let SIM =
(* The users biometric data and the users text are input to the SIM safely via private channels.*)
in(c1,nx);
in(userChBD,BD);
in(userChText,userText);
(* The received biometric data is encrypted using the SKCG, the session key
which is shared between the SIM and the user card *)
let BDsenc = senc(BD,sSKCG) in
out(c2,(BDsenc,h((sSKCC,nx),BDsenc)));
in(c3,m20);
(* When the SIM receives the acknowledgement that SMC satisfies with the
freshness of the biometric data that it receives, the SIM output the user
command i.e. sign the users text using the users private key *)
if m20 = OK then
( out(c4,userCommand);
(* The SIM receives the messages from the SMC and send them to the user
card in order to acquire the user card to verify the users biometric data
against the stored biometric code *)
in(c5,(m4,m5));
out(c6,(m4,m5));
in(c7,(m6,m7));
out(c8,(m6,m7));
in(c9,okm);
(* If the biometric verification succeeds, the users document is encrypted
and the hash value of the encrypted data is generated. The message is sent to the SMC. *)
if okm = OK then
( let digest = senc(hashDoc(userText),sSKCG) in
out(c10,(digest,h(sSKCC,digest)));
in(c11,(m13,m14));
out(c12,(m13,m14));
in(c13,m15)
)).
```

Figure 4.6: SIM Process for the Analysis of [7]

```
let UserCard =
(* The SMC perform mutual authentication with the user card;
the authentication method is not specified. In this case, we have used
Needham Schroeder Lowe Protocol in order to perform the authentication *)
in(c,m);
let(NY,hostY) = dec(m,skUserCard) in
let hostUserCard = host(pk(skUserCard)) in
out(c,(hostUserCard,hostY));
in(c,ms);
let(pkY,=hostY) = checksign(ms,pkS) in
new Nb;
out(c,enc((NY,Nb,hostUserCard),pkY));
in(c,m3);
if Nb = dec(m3,skUserCard) then
let skcg = h(NY,Nb) in
let skcc = g(NY,Nb) in
let ssc = f(NY,Nb) in
(* After the mutual authentication succeeds, the user card receives the
request message to perform the biometric verification. If the biometric data
matches with the stored biometric template, the matching result is output*)
in(c6,(m10,m11));
if h((skcc,ssc),(userCommand,m10)) = m11 then
( let BDsdec = sdec(m10,skcg) in
if BDsdec = BD then
let SW = success in
let m12 = h((skcc,ssc),SW) in
out(c7,(SW,m12));
in(c12,(m18,m19));
(* The user card checks the validity of the users document. The
satisfaction leads the user card to release the stored users private key for signing the document. *)
if h((skcc,ssc),m18) = m19 then
( let M2 = sdec(m18,skcg) in
out(c13,sign(M2,skUserCard))) ).
```

Figure 4.7: UserCard Process for the Analysis of [7]

```
let U =
(* The user is presented the document. If the user satisfies with the
presented document, the user will present his biometric data *)
in(TextCh,t);
if t = Text then
out(userChBD,BD);
out(userChText,t).
```

Figure 4.8: User Process for the Analysis of [7]

```
let S =
(* The server is assigned the public key to the identity *)
in(c,m);
let(a,b) = m in
let sb = getkey(b) in
out(c,sign((sb,b),skS)).
```

Figure 4.9: Server Process for the Analysis of [7]

```
Process
(* The communication channels are created *)
new userChBD; new userChText; new AliceTextCh; new BobTextCh;
(* The biometric data of the legitimate user, Bob, and the biometric data
of an attacker, Alice, are input in the channel *)
new BobBD; new AliceBD;
(* Session keys, publics key and private keys are created *)
new sSKCG; new sSKCC; new skSMC;
let pkSMC = pk(skSMC) in
out(c,pkSMC);
new skBobUserCard;
new skAliceUserCard;
let pkBobUserCard = pk(skBobUserCard) in
let pkAliceUserCard = pk(skAliceUserCard) in
out(c,pkBobUserCard);
out(c,pkAliceUserCard);
new skS; let pkS = pk(skS) in
out(c,pkS);
(* Bob and Alice try to present the documents that they wish to sign and
their biometric data to the system *)
!out(AliceTextCh,AliceText);
!out(BobTextCh,BobText);
((!S) | (!SMC) | !SIM |
(let TextCh = AliceTextCh in
let Text = AliceText in
let BD = AliceBD in !U) |
(let TextCh = BobTextCh in
let Text = BobText in
let BD = BobBD in !U) |
(let skUserCard = skAliceUserCard in
let BD = AliceBD in !UserCard) |
(let skUserCard = skBobUserCard in
let BD = BobBD in !UserCard))
```

Figure 4.10: Main Process for the Analysis of [7]

Chapter 5

Attestation-Based Remote Biometric Authentication

In chapters 3 and 4, we introduced two different biometric authentication protocols: one for generic use and the other one for a specific purpose, signing a document. However, the basic scenario for both protocols is the same: local biometric authentication. That is, a user presents his biometric data to the local biometric reader and the biometric verification is performed locally, within the computing platform [23] or within the user card [7].

In [23], the protocol is based on the assumption that the biometric reader has extended capability, enabling it to generate a session key in order to decipher user biometric data read from the trusted biometric reader before it is transmitted. The biometric reader is verified by the TPM if it is trusted. In addition, the protocol assumes that the TPM has extended capability, enabling it to perform biometric matching.

In contrast with the above protocols, [17] presents a protocol that does not require additional functionality of the components involved in the protocol. The protocol simply uses the signature of the TPM to ensure validity of the biometric data, and the signature of the biometric matching server is used to guarantee validity of the matching result. This protocol relies on an SSL channel in communicating with the public channel.

This chapter illustrates an attestation-based remote biometric authentication protocol. The protocol is presented by Polon and Sander [17]. The protocol is designed for remote login using a user's biometric data. This remote biometric authentication protocol allows a user to validate his identity by using his biometric feature remotely. The biometric feature is input via a local client while the biometric verification takes place remotely through the biometric matching server. The result is then transferred to the server which the user intends to login to. The physical setup of this protocol is shown in figure 5.1. A detailed description of the protocol is presented in a later section. The verification and analysis of this protocol are illustrated in the following section.

5.1 The Protocol

The [17] protocol is designed to allow a user to remotely login to the remote service using his biometric data. The protocol uses the TPM for the integrity checking. The TPM contains sixteen Platform Configuration Registers (PCRs). These PCRs are used for storing integrity-measured value when a component is loaded. If it is reloaded, the value is measured and rechecked with the previously stored value. If it differs, the appropriate action will be taken. The next layer will not be loaded unless the previous layer has been verified. The protocol consists of three main components: *Biometric Authentication Server* (BAS), *remote service*, and *local client*. The BAS is a server which is responsible for authentication of the user through a biometric matching process. The remote service is a server that services the user, e.g. enables user login. The local client is a computer which is used by the user. To increase security, this research uses a laptop computer which has a biometric reader and TPM installed. The principal software installed in the local client consists of an operating system, network client, biometric software such as

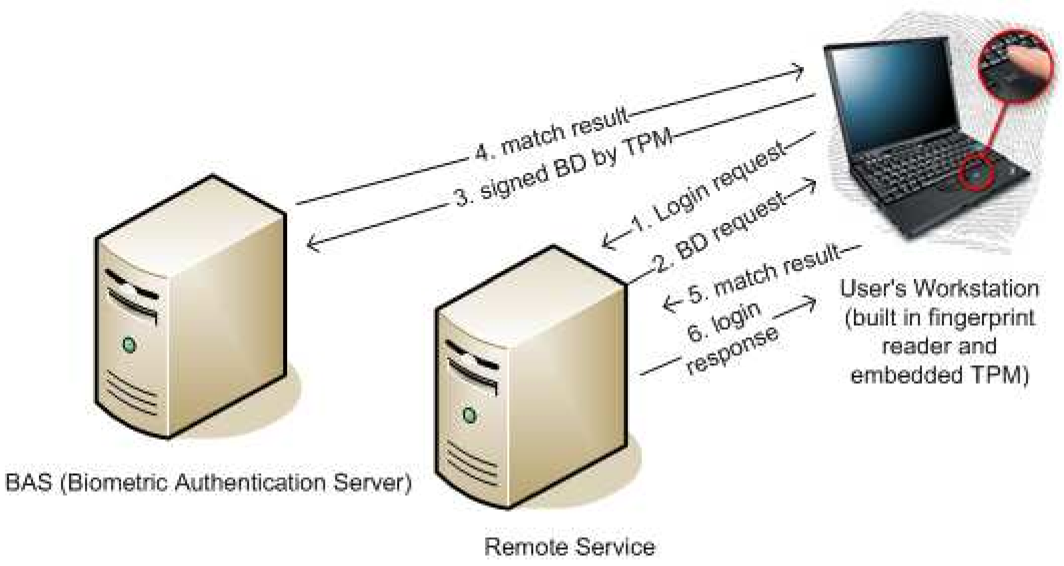

Figure 5.1: The Physical Setup of [17]

BioAPI (Biometric Application Programming Interface), and a biometric device driver. The BioAPI enables interoperability and integrity in systems that intend to use biometric technologies. In this paper, the BioAPI interacts with the TPM and the BAS. The TPM is used for integrity checking to ensure that the platform has not been altered. The TPM also signs the biometric data in order to guarantee its origin to the recipient. The BioAPI sends the BAS the captured biometric data so that the server can verify the user's identity.

The protocol starts when a user wishes to login to the remote service, he sends a login request to that server. As a response, the server requests an authentication challenge which triggers the local client to acquire the user's biometric data. Specifically then, the local client sends this authentication challenge to the BioAPI which in turn triggers the biometric driver to read the user's biometric data from the biometric reader. When the user places his finger on the fingerprint sensor which is installed in the computer workstation, this process produces biometric data (BD) as shown in the message sequence chart (figure 5.2). The biometric data is sent back to the BioAPI via the biometric driver. The BioAPI requests a signature from the TPM for the biometric sample. The TPM is embedded in the workstation. It signs the biometric data and sends it back to the BioAPI. The BioAPI forwards the signed biometric data to the BAS. The BAS is responsible for verifying the user's biometric data. Before the biometric verification is performed, the signed biometric data must be validated. For the sake of the signature verification, the BAS registers each user's workstation beforehand so that it knows the TPM's key. The BAS verifies the TPM's signature. If it is valid, the BAS checks the biometric data against the stored biometric code. The matching result is generated and the response is signed by the BAS and sent to the BioAPI. In this research, the BioAPI interacts with the BAS and the TPM. In this protocol, the BioAPI involved in the protocol has enhanced capability beyond the current BioAPI specification in [34]. The local client on behalf of the BioAPI forwards the biometric verification response to the remote service. The remote service verifies the signature, checks the authentication result, and sends back the login result: whether to allow or refuse login. The communication pathway associated with this description is shown in figure 5.2.

5.2 Verification of the Protocol

ProVerif is used for verifying this protocol. This section is divided into 4 subsections: *signature and equational theory*, *workstation* process, *remoteService* process, and *BAS* process.

5.2.1 Signature and Equational Theory

The *sign* function is used to sign the message whereas the *pk* function is used to generate the public key. The `equation checksign(sign(x,y),pk(y)) = x.` is represented for verifying the signature of a message from the provided public key of the origin. ProVerif code for the signature and equational theories is presented in figure 5.3.

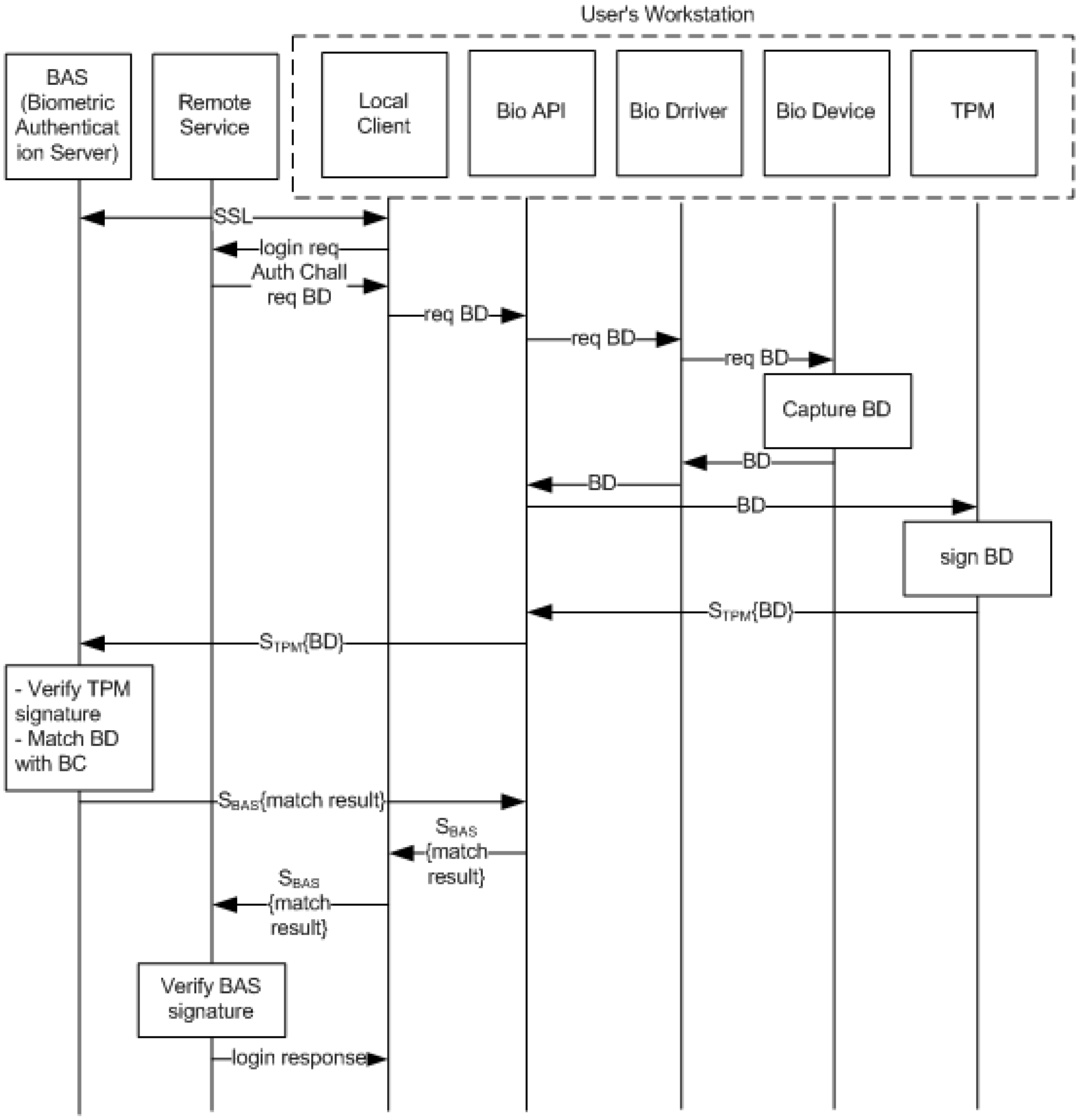

Figure 5.2: The Communication Messages for the Remote Biometric Authentication [17]

5.2.2 Workstation Process

The workstation process sends a login request to the channel. The biometric data is created as the user places his biometric data on the biometric reader. The biometric data is signed by the TPM's private key and sent out. The result of the matching process is received and forwarded to the remote service. The workstation is waiting for the login result. The ProVerif model of the workstation interpretation is presented in figure 5.4.

5.2.3 remoteService Process

Upon acceptance of the login request, the remote service requests authentication from the user. As a result, the authentication challenge is sent out to the local client which then triggers the BioAPI to acquire the biometric data from the reader. The matching result is received and the signature verified. If the user's verification is successful and the biometric data is matched, the remote service outputs the login response and transmits it to the client. The ProVerif model can be found in figure 5.5.

5.2.4 BAS Process

In the BAS process, biometric verification is performed and the verification result sent out. Normally, the biometric matching server will wait for the biometric data and then perform the matching process of the biometric data against the stored biometric code. Once it is received, the signature of the TPM is verified. The biometric data is matched against the stored biometric code. The matching result is created, signed by the BAS's private key, and sent out. As shown in figure 5.6, the public key of the biometric authentication server is created and sent out via the private channel. This public key will be received securely by the remote service so that it can be used for deciphering biometric matching result.

5.2.5 Alice Process

The Alice Process represents an intruder who sits between the local client and the Biometric Authentication Server. The ProVerif model of Alice is presented in figure 5.7. Alice tries to intercept the matching result which is sent along the channel and she knows the private and public key pairs in the SSL channel.

Alice could try many times until she gets a positive login response.

5.2.6 Main Process

The *privCh* and *pkBASch* are set up as private channels. The *privCh* is used for distributing the public key of the TPM while the other channel is used for distributing the BAS's public key. Therefore, these two keys are received securely. The main process replicates and runs the four processes *workstation*, *remoteService*, *BAS*, and *Alice* concurrently. The matching result is input in the BAS process in order to verify one of the properties of the protocol, *authenticity*. The ProVerif model is shown in figure 5.8.

5.3 Analysis of the Protocol

We analyse the protocol to verify the properties that should hold. The privacy of the biometric data is analysed to ensure that the user's data is not revealed to an intruder. The authenticity property is analysed to verify whether an attacker can capture the biometric data verification result and use it as his own result to successfully login to the remote service himself.

5.3.1 Privacy of Biometric Data

The privacy property is verified to check whether an attacker can hold the biometric data. As a consequence, he could login as a legitimate user. Although, we consider that the biometric data is in the public domain, the privacy property is desirable. This property is indeed required to be verified (as we verify this property in the other protocols). The *query* is executed to verify this property. ProVerif executes this command to check whether an intruder could acquire the biometric data, BD.

```
query attacker : BD.
```

The result of the verification is positive. The biometric data is kept private within the protocol.

5.3.2 Authenticity

This protocol is proposed to enable remote login by a user. Therefore, the authenticity property is analysed to check whether an intruder could retrieve the biometric matching verification and later use it to login as a legitimate user. We model the intruder's insertion of the captured success between the *local client* and the *remote service* as matchResult. We assume that an intruder obtained the public/private key pair which is used in the SSL channel between the *local client* and the *remote service*.

The ProVerif analysis is :

```
query attacker :  AliceLogin.
```

The verification result shows that an intruder could not gain the positive *matchResult* and he can not later insert this result into the protocol. This analysis shows that the protocol does not allow an intruder to login as the legitimate user.

5.4 Chapter Summary

This chapter presents a remote biometric authentication protocol [17] for login to a remote server. The protocol is modelled in ProVerif and verified. We have analysed two properties, *privacy of biometric data* and *authenticity*. The positive result for both properties shows that the protocol satisfies the two properties.

As the protocol uses biometric data for user verification, the privacy of biometric data is desirable. The results from the analysis show that the protocol holds this intended property.

The authenticity property of the protocol is analysed to verify that the protocol performs its function of remote login effectively.

```
(* SIGNATURE *)
fun pk/1.
fun sign/2.
fun checksign/2.
(* EQUATION *)
equation checksign(sign(x,y),pk(y)) = x.
```

Figure 5.3: Signature and Equational Theory for the Analysis of [17]

```
let workstation =
(* The user received the request of the users biometric data in order to verify the user *)
in(c,reqBD);
new BD;
out(c,sign(BD,skTPM));
in(c,m3);
out(c,m3);
in(c,m5);
if m5 = allow then
out(c,successLogin).
```

Figure 5.4: Workstation Process for the Analysis of [17]

```
let remoteService =
in(pkBASch,pubBAS); (* The remote service receives its public key via private channel *)
(* When the remote service receives the request to login to the specific
application, the remote service requests the user authentication challenge*)
in(c,req);
out(c,authChall);
in(c,m4);
(* The remote service checks the biometric verification result. If the
verification is successful, the remote service allows the user login *)
let matchResultReceived = checksign(m4,pubBAS) in
if matchResultReceived = match then
let response = allow in
out(c,response).
```

Figure 5.5: RemoteServcie Process for the Analysis of [17]

```
let BAS =
(* The biometric authentication server deciphers the encrypted biometric data.
The matching result is produced and sent to the channel *)
new skBAS;
let pkBAS = pk(skBAS) in
out(pkBASch,pkBAS);
in(c,m);
let BDReceived = checksign(m2,pkTPM) in
out(c,sign(matchResult,skBAS)).
```

Figure 5.6: BAS Process for the Analysis of [17]

```
let Alice =
in(c,m7);
out(c,m7);
in(c,m8);
if m8 = allow then
out(c,AliceLogin).
```

Figure 5.7: Alice Process for the Analysis of [17]

```
process private free privCh,pkBASch.
new skTPM;
new successLogin; (*Login response for the legitimate user *)
new AliceLogin; (*Login response if an intruder, Alice, can login *)
new match; (*Matching result if BD and BC match*)
new noMatch; (*Matching result if BD and BC does not match *)
new allow; (*Allowance response if the remote service allow
a particular user to login *)
let pkTPM = pk(skTPM) in
out(ch,pkTPM);
!workstation | !remoteService |
(let matchResult = match in !BAS)
(let matchResult = noMatch in !BAS) | !Alice
```

Figure 5.8: Main Process for the Analysis of [17]

Chapter 6

A Remote Biometric Authentication Protocol for On-line Banking

This chapter introduces a remote biometric authentication protocol for on-line banking system. On-line banking is an example of non-supervised biometric authentication as the system authenticates the user remotely. In a non-supervised biometric authentication, the authentication process is not controlled by the verifier.

Biometric security depends on the authenticity of the biometric data. As discussed, the biometric data is in the public domain by its nature and artificial biometric data can often fool a biometric reader. Biometrics is hard to keep secret. Human has a limited number of them and they can not be changed. Biometric authentication is much harder in the remote or unattended cases. Liveness detection in biometric reader is largely research. Biometric authentication works well in supervised situations but for high assurance situations, the reader should be attended to or at least observed until we get verifiably strong liveness detection. However, this concern mainly relies on research in hardware.

This chapter presents an approach to enhance security level for a remote biometric authentication. The protocol guarantees the live presentation of the user on the time of verification, liveness property. On-line banking is employed as an example for illustration the proposed protocol. This chapter attempts to demonstrate a remote biometric authentication that assures the liveness property; therefore we assume that the related risks of the on-line transaction e.g. key logger or viruses is out of the scope.

6.1 The Protocol

The proposed protocol is an on-line banking that requires biometric authentication. The user is requested to authenticate to the system using his biometrics, e.g. his fingerprint, before he is allowed to proceed with the transaction. In this protocol, transferring money between accounts is used as an example of an on-line banking transaction.

The scenario of the protocol can be illustrated as a user wishes to transfer money from his account to another account. The bank asks for the user's authentication. This requires the user to present his biometric data. The biometric data is sent to the authentication server which responsible to perform biometric verification.

The result from the verification is supplied to the bank, and so and it will decide, upon this result, whether to allow the user to carry out the transaction he requests. If the biometric verification is positive, it triggers the bank to perform the user's request, and transfer money from this account to his desired account.

The proposed protocol consists of three components: the user, the bank and the biometric authentication server. The user requests the transfer transaction. He provides his biometric data for the user's authentication. The user operates his transactions via the local workstation which has TPM installed. Upon booting up, the user's workstation and the biometric reader are verified by

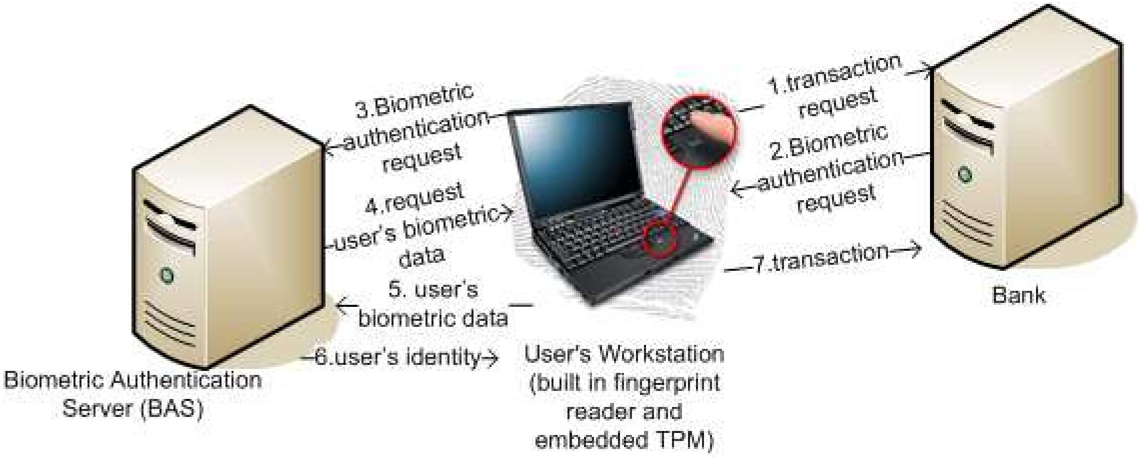
3.Biometric authentication request
4.request user's biometric data
5. user's biometric data
6.user's identity
1.transaction request
2.Biometric authentication request
7.transaction
Bank
Biometric Authentication Server (BAS)
User's Workstation (built in fingerprint reader and embedded TPM)

Figure 6.1: The Physical Setup of the Remote Biometric Authentication Protocol for Online Banking

the TPM. The report of the integrity of the local system is sent to the user. If he is satisfied with the integrity of the system, he continues to proceed with his activities. The bank is responsible for the bank transaction which is transferring money as requested by the user. The biometric authentication server performs the biometric matching process. It has biometric template storage which is used when the biometric code is acquired for matching against presented biometric data. It reports the user's verification result to the user. The user presents the verification result along with this request transaction to the bank. The physical setup of the protocol is presented in figure 6.1.

The protocol involves two major activities: authenticating the user, and transferring money, each of which has different consideration. A user is required to verify himself to the bank to in order to access his account. As the authentication process requests the user's biometric data, the *liveness of the biometric data* must be assured so that the bank is certain that he is an authentic user and he is willing to provide his biometric data for the transaction.

Once the user's authentication is successful, the user requests money transfer transaction to the bank. The bank checks the validity of the user's verification result. If it is valid and the biometric verification is positive, the bank executes the requested transfer transaction. The *intensional authentication* property is considered. The communication messages of the protocol is shown in figure 6.2.

The communication messages commence when the user requests transfer transaction from the bank. As a response, the bank looks for the user's authentication, in this case the biometric authentication is applied. The bank sends a signed message which includes the user name, the biometric authentication request and the nonce n1. The user forwards this message to the biometric authentication server in order to acquire the user's verification. It triggers the BAS to inquire the user's biometric data; the nonce n2 is included in the replied message. This requested message is signed with the BAS's signature. The user presents his biometric data to the reader. The biometric data is signed by the TPM so that the authenticity of the biometric data can be verified. The encrypted message is composed of user name, the nonce n2 and the signed biometric data with the TPM's signature.

To enhance the security requirement in term of liveness of the biometric data, once the biometric data is submitted to the BAS, the BAS verifies the live presentation of the user by acquiring the user to present verification data. The verification data is a secret data which is known only to him and the BAS. The BAS sends an encrypted message of the request and newly generated nonce n3. As a response, the user presents his biometric data, the verification data and the nonce n3. The message is signed with the TPM's private key. The signed message, together with information the user provided, are enciphered by the BAS's public key. The BAS verifies the message by checking the nonce and the signature. It then validates the authenticity of biometric data. The verification result, the user name and the nonce n1 are signed by the BAS.

Upon receiving the verification result message, the user appends his transfer transaction and sends this message to the bank. The transfer message includes the amount and account he wishes to transfer. The message is encrypted by

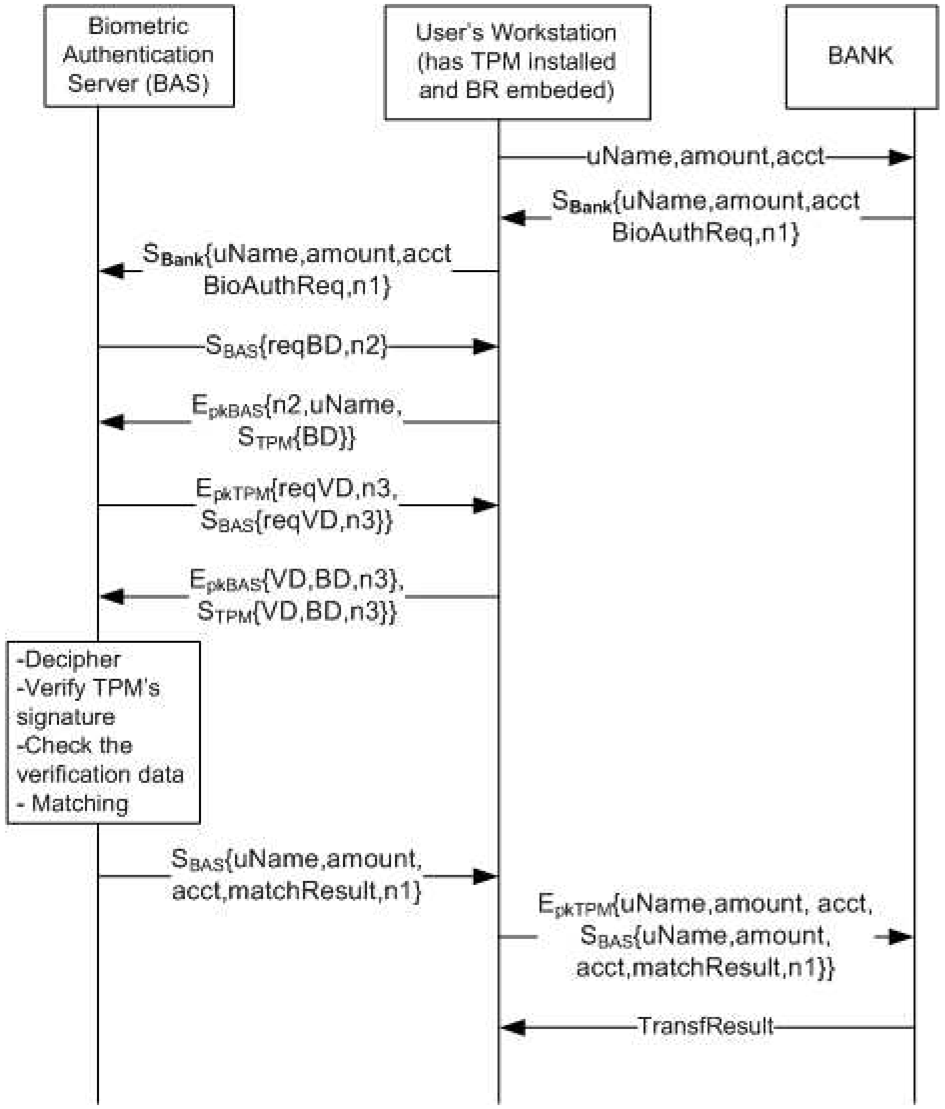
Biometric Authentication Server (BAS)
User's Workstation (has TPM installed and BR embeded)
BANK
uName,amount,acct
S_{Bank}{uName,amount,acct BioAuthReq,n1}
S_{Bank}{uName,amount,acct BioAuthReq,n1}
S_{BAS}{reqBD,n2}
E_{pkBAS}{n2,uName, S_{TPM}{BD}}
E_{pkTPM}{reqVD,n3, S_{BAS}{reqVD,n3}}
E_{pkBAS}{VD,BD,n3}, S_{TPM}{VD,BD,n3}}
-Decipher
-Verify TPM's signature
-Check the verification data
- Matching
S_{BAS}{uName,amount, acct,matchResult,n1}
E_{pkTPM}{uName,amount, acct, S_{BAS}{uName,amount, acct,matchResult,n1}}
TransfResult

Figure 6.2: The Communication Messages for the Remote Biometric Authentication for On-line Banking

the public key of the bank. The bank deciphers and verifies the validity of the message. It then checks the matching result. If the result is positive, the bank performs the user's request, transferring the money to his desired account.

6.2 Protocol Properties

This remote biometric authentication protocol for on-line banking has three intended security properties: *privacy of biometric data*, *liveness* and *intensional* *authentication.*

- Privacy of biometric data. The biometric data is kept private within the protocol. It is a good practice for the protocol to consider the biometric data not to be spread around without restriction. The protocol prevents the entities which are not included in the protocol to obtain the biometric data.
- Liveness. The authentication process will only perform on the biometric data that comes from the live presentation of the user. In order to achieve this property, the protocol provides an approach to request the verification data from the user when he is presenting his biometric data. The verification data is known only to the user. If the data can be supplied during the biometric authentication process, it could be guaranteed that the biometric data presented is not from the artificial biometric data. The liveness property is claimed.
- Intensional authentication. The protocol transfers the correct amount to the correct account. This protocol can prevent an intruder to intercept the message and transfer to her account or other account that the legitimate user does not wish to.

6.3 ProVerif Model

The proposed protocol promises the three intended properties. To ensure that the protocol provides the appropriate level of security and the properties it affirmed, ProVerif is used as the verification tool to verify and analyse the protocol. The ProVerif models consist of three major processes: BAS process, Workstation process and the Bank Process. The BAS process represents the duties of the BAS. It's main task is to authenticate the user and ensure the live presentation of the biometric data. The workstation process represents the user's activities: presenting the biometric data for the biometric authentication and request transfer transaction to the bank. The U process is responsible to input the user's verification data and user's biometric data to the workstation. The Bank process represents the bank's business. It verifies the biometric authentication result and performs the transfer transaction as requested by the user if the authentication is successful. The ProVerif models of the protocol are described in the following section.

6.3.1 Equational and Signature Theory

This is the method that ProVerif uses to solve the messages. From the equational and signature theory (shown in figure 6.3), ProVerif also forms messages from the provided theory in order to solve whether an attacker can acquire the information that the *query* commands ask for. It, in turn, returns the verification results. The *checksign* operation is performed in order to verify the signature of the message and the information in the signed message is presented. The signed message is performed through *sign*. The *enc* is used for message encryption. The messages are encrypted by public key cryptography. To decipher a message, *dec* is performed to decipher an encrypted message from the known key.

6.3.2 BAS Process

When the BAS process receives the authentication request, the BAS responds the request by sending a signed message of newly generated nonce n2 and biometric authentication request. It then awaits for the biometric data to be sent to. In order to verify the live presentation of the user, the BAS sends the message that acquires the verification data. This message includes the newly generated nonce n3 and verification data request. The message is signed by the bank to specify the origin of the message. To secure the message and the validation checking purpose, the nonce, verification data and the signed package are encrypted by the public key of the TPM. The BAS expects to receive the biometric data and secret verification data sent from the user's workstation. Upon receipt, it deciphers the message, checks the validity of the data and performs the user's biometric verification. The matching result, user name and the nonce n1 which is received when the BAS has accepted the authentication result are signed by the BAS so that the recipient can validate the origin of the message.

6.3.3 Workstation Process

The user's workstation initiates the communication by sending a transfer request to the bank. It then receives a request for biometric authentication and the nonce n1 from the bank. The workstation process forwards this message to the BAS. It then receives the authentication challenge and the nonce n2 from the BAS. The user places his biometric data on the sensor. This results in generation of the biometric data in ProVerif as shown in figure 6.5. His biometric data is signed by the TPM to guarantee it origin. The username, the nonce n2 and the signed biometric data are encrypted by the BAS's public key. This message is supplied to the BAS. The workstation process receives the request for the verification data. It then obtains this data from the user and sends it out. It expects to receive the biometric matching result. The workstation process performs the transfer transaction by sending the relevant information to the bank in encrypted format. The transfer transaction is sent out together with the authentication result.

6.3.4 Bank Process

The Bank process represents the transfer transaction. Let us describe the process presented in figure 6.6, the bank first receives the transfer request from the user or local's workstation. As a response, it then generates a nonce n1 and sends it with the authentication request. The bank waits for the reply message which is expected to be an authentication result. The message the bank receives is encrypted by the public key of the TPM. It then deciphers the message; it checks the validity of the message by checking whether the first nonce is the same as the one it sent to the workstation. If so, the bank checks the signature of the message by checking whether it came from the BAS. The bank checks the matching result. If the result is positive, the bank then performs the transfer transaction.

6.3.5 U Process

The U process is awaiting for the request from the workstation to present his biometric data and verification data. The ProVerif model of the U process is shown in figure 6.7.

6.3.6 Main Process

The BAS process, the Bank process the user's workstation process and the U process are replicated and executed in the main process (figure 6.8). The main process generates the keys for each process, these keys including public/private key pairs of the BAS, TPM and bank.

6.4 Analysis of the protocol

The thesis analyses the three intended properties of the protocol. The protocol commits to provide *privacy of biometric data*, *liveness* and *intensional authentication* each of which is verified and analysed as shown in the following.

6.4.1 Privacy of Biometric Data

The protocol should be analysed so that it does not have risk of spreading around the user's biometric data. The biometric data used in the protocol should not be revealed to entities which are not included in the protocol. The *query* command in ProVerif is executed to determine whether an attacker could intercept the data successfully.

```
query attacker :  BD
```

6.4.2 Liveness

The protocol guarantees that it can deny access from the artificial biometric. Therefore, the liveness property must be verified. The following query tries to analyse whether Alice, an intruder, holding an artificial biometric data e.g. rubber finger of legitimate user, Bob, can obtain the positive authentication result as if she is Bob.

```
query attacker :  AliceName.
```

6.4.3 Intensional Authentication

Since the protocol is used for a particular purpose, money transfer, the intensional authentication of this protocol can be illustrated as the money is transfer to the correct account for the correct amount as the user wishes. The analysis of the intensional authentication property for this protocol refers to whether an attacker can capture the authentication result and use it to transfer money by posing as the legitimate user. If the bank process exposes AliceAcct to the public channel, the attack is found. The ProVerif analysis is shown as below:

```
query attacker :  AliceAcct.
```

The verification result illustrates that the protocol is secure for use in on-line money transfer in a biometric authentication situation. Alice cannot intercept and manipulate the transfer message to transfer the money to her account.

6.5 Chapter Summary

This chapter proposes a remote biometric authentication protocol. The protocol uses on-line banking to illustrate the protocol. The protocol provides an approach to verify the liveness of the biometric data. The user has to provide his secret verification data in order to prove that he is presented on the time of biometric authentication. The protocol uses the TPM to verify the user's workstation and the biometric reader. This enhances the security level of the protocol. The user is assured that the workstation and the reader he is using will not manipulate his data and request. The signatures are used to guarantee the origin of the messages. The public key encryptions are applied to secure the messages. The proposed protocol guarantees three properties: *privacy of biometric data*, *liveness* and *intensional authentication*. The positive results from the verification show that the protocol holds the three security requirements.

The next chapter sums up what we have studied in the thesis. From our investigation and verification, we propose properties that are desirable of a biometric authentication protocol. These properties could be used as guidance when proposing a new biometric authentication protocol.

```
(* SIGNATURE *)
fun sign/2.
fun checksign/2.
fun pk/1.
fun dec/2.
fun enc/2.
```

```
(* EQUATION *)
equation checksign(sign(x,y),pk(y)) = x.
equation dec(enc(x,pk(y)),y) = x.
```

Figure 6.3: Signature and Equational Theory for the Analysis of Remote Biometric Authentication Protocol for On-line Banking

```
let BAS =
in(c,(uyName,amounty,accty,BioAuthReqx,nxx1));
new n2; out(c,sign((reqBD,n1),skBAS);
in(c,m11);
let (=n2,=uyName,signBD) = dec(m11,skBAS) in
let (BDReceived) = checksign(signBD,pkTPM) in
new reqVD;
new n3;
let m12 = sign((reqVD,n3),skBAS) in
let m13 = enc((reqVD,n3),pkTPM) in
out(c,(m12,m13));
in(c,(m14,m15));
let (VDReceived,=BDReceived,=n3) = dec(m12,skBAS) in
let (=VDReceived,=BDReceived,=n3) = checksign(m15,pkTPM) in
let matchresult = ok in
let m16 = sign((uyName,amounty,accty,matchresult,nxx1) in
out(c,m16);
```

Figure 6.4: BAS Process for the Analysis of Remote Biometric Authentication Protocol for On-line Banking

```
let WorkStation =
in(c,uName,amount,acct) out(c,uName,amount,acct);
in(c,(BDReqx,nx2));
out(c,BDReqx);
in(c,BDin);
out(c,enc(nx2,uName,BDin),pkBank));
in(c,m3);
let m4 = dec(m3,skTPM) in
let (reqVD,nx3) = checksign(m4,pkBank) in
out(c,(reqVD,nx3));
in(c,VDin);
out(c,enc((VDin,BDin,nx3,sign((VDin,BD,nx3),skTPM)),pkBank));
in(c,matching).
let m10 = enc((uName,amount,acct,matching),pkBank)
in(c,transfResult);
```

Figure 6.5: WorkStation Process for the Analysis of Remote Biometric Authentication Protocol for On-line Banking

```
let Bank =
in(c,(uxName,amountx,acctx));
new n1;
new BioAuthReq; out(c,sign((req,BioAuthReq,n1),skBank);
in(c,m1);
let (=n1,=uxName,=amountx,=acctx) = dec(m1,skBank) in
let (=uxName,=amountx,=acctx,matchresultreceived,=n1) = checksign(m2,pkBank) in
If matchresultreceived = ok then
out(c,acctx);
```

Figure 6.6: Bank Process for the Analysis of Remote Biometric Authentication Protocol for On-line Banking

```
let U =
in(c,(n,amt,acc));
out(c,(name,amt,accout));
in(userVD,VDq);
out(userVD,VD).
```

Figure 6.7: User Process for the Analysis of Remote Biometric Authentication Protocol for On-line Banking

```
process
new skTPM;
let pkTPM = pk(skTPM) in
new skBAS;
let pkBAS = pk(skBAS) in
new skBank;
let pkBank = pk(skBank) in
out(ch,pkTPM);
out(ch,pkBAS);
out(ch,pkBank);
!WorkStation | !Bank | !  BAS |
(let name = AliceName in
let amout = AliceAmount in
let account = AliceAcct in |U) |
(let name = BobName in
let amout = BobAmount in
let account = BobAcct in |U)
```

Figure 6.8: Main Process for the Analysis of Remote Biometric Authentication Protocol for On-line Banking

Chapter 7

Conclusions

In this final chapter, we review the thesis, draw appropriate conclusions and suggest areas for further work. First, we presented an existing biometric authentication protocol and the nature of biometric data that might be used in biometric authentication. Then we studied verification methods and the desirable properties of biometric authentication protocols. A remote biometric authentication protocol was then considered, followed by an on-line banking case study.

7.1 What We Have Learnt

Firstly, in order to understand biometric authentication, the nature of biometric data was investigated. When biometric data is used in an authentication protocol, it needs to be treated differently from other types of data. Biometric data is in the public domain, whereas authentication data such as a password is not. Therefore, security considerations are different.

In chapter 1, we considered why, although biometric data is not secret, its use should not be widespread without restriction. The importance of verification of security protocols was also introduced. This led to the work in the following chapters.

The biometric authentication protocols to be studied in this thesis were then selected. The two protocols are used for different purposes. Therefore, the intended properties of the protocols are different. The CPV02 protocol is for generic biometric authentication and can be extended to include, for example, release of the user's private key for signing a document if user authentication succeeds. This protocol uses a TPM, an important component in the system. Before any component is given access to the biometric data or code, the TPM examines that component to check whether it can be trusted.

Next, a biometric authentication protocol for a particular application was presented: to create a signature to sign a document [23]. The thesis extended the signature creation part of this protocol in order to verify one of the properties.

Desirable properties of these two protocols tend to differ, but one common property that we wished to verify was that of privacy of biometric data. Although biometric data is in the public domain, when it is captured by the biometric reader, it should not be leaked in this format to a third party outside those involved in the protocol. This is what we mean when we refer to the property of privacy.

In chapter 2, we introduced the concepts of the trusted platform module (TPM) and the Dolev Yao attacker. The TPM plays an important role in the CPV02 protocol. It is responsible for checking the integrity of the components involved in biometric authentication, which include the computing platform and the biometric reader. The computing platform carries out biometric matching. Therefore, it must be trusted in terms of integrity checking by the TPM before the biometric data and biometric code are transferred to it. Similarly, the biometric reader must be trusted before it is allowed to receive the biometric data from the user.

The Dolev Yao attacker is important in the verification process of the protocols. This powerful attacker, which can interfere with channels, capture messages, and regenerate an encrypted message from a known key, is significant when the protocols are modelled in ProVerif. Modelling this type of powerful attacker yields robust verification results.

Study and analysis of the CPV02 protocol showed that naive interpretation of the protocol produces negative verification results. However, clarification of the protocol enabled us to analyse three properties of the protocol, which were all shown to hold. Analysis and verification of this protocol was valuable. Not only did it enable us to consider the function and nature of the TPM, but detail in chapter 3 also demonstrates how the TPM can provide a high level of security.

Analysis of the biometric authentication protocol for a signature creation application in chapter 4 employs an attacker that could occur in the local machine. The section on a chip and pin interceptor utilises the Dolev Yao style attacker in such a way that an attacker not only intercepts messages in the public channel but also captures the communication signal in the wired connection. This chapter models an attacker that tries to interfere with the communication message between the user and the SIM, analysing whether an intruder could intervene and lead the user to sign the intruder's document. Analysis of two properties is presented: *privacy of biometric data* and *intensional authentication*. As mentioned earlier, the privacy property is desirable even though the data is already considered to be in the public domain. Verification shows that this property holds. In the case of *intensional authentication*, the verification result shows that an intruder could not interfere with the communication channel to obtain a counterfeit signature from the user.

An interesting protocol which claims that it does not require either the biometric sensor to generate a random number as in [4], or the TPM to carry out the matching process as in [23], is presented in chapter 5. This chapter first introduces a remote biometric authentication protocol. The verification of the protocol is applied. We have learnt the without the protocol extension,

Table 7.1: Summarisation of the properties that each protocol achieves

Protocol / Property	Effectiveness	Correctness	Privacy	Intensional	Liveness
On Enhancing Biometric Authentication with Data Protection	YES	YES	YES	NO	NO
Protecting Transmission of Biometric User Authentication Data for Oncard-Matching	No	NO	YES	YES	NO
Attestation-Based Remote Biometric Authentication	NO	NO	YES	YES	NO
Remote biometric authentication protocol for On-line Banking	NO	NO	YES	YES	YES

one of the intended properties cannot be successfully analysed.

We have found that the *liveness* property is important, especially when the protocol is used in a non-supervised situation, we propose a new remote biometric authentication protocol for on-line banking in order to serve this property. The proposed protocol uses a transfer transaction as an example of the transaction to be performed by the bank. In section 6, the protocol is described, and we verify the protocol and its intended properties. Analysis of the protocol showed that the desired properties hold for this protocol.

Investigation during analysis of protocols in this thesis shows that the protocol should be clearly described in terms of its intended purposes, the sequences of the protocol and mandatory requirements. Without these clarifications, the protocol would not be successfully analysed. Table 7.1 shows the summarisation of the properties that each protocol tries to achieve.

The protocols are differently modelled and achieve different security requirements; attacks to the protocols are different, therefore verifications and analysis of protocols are district. Some properties are generated specifically for the protocols. Therefore, a generic approach for analysis cannot possibly be generated. Even the protocols hold the same property, the analysis is different and due to the attacks to the protocol are different.

In this thesis, we have investigated three case study protocols and analysed them. The detail of the verification can be applied for use in analysing other protocols that offer the same properties and purposes.

We have proposed a new remote biometric authentication protocol that guarantees the safety use of biometric data in a non-supervised situation.

During analysis processes, we have discovered the desirable properties that each protocol should hold. Protocols that offer different purposes provide different properties. Theses desirable properties and their descriptions are useful for designing and developing biometric authentication protocols.

The privacy property is crucial for a biometric authentication protocol. It should not be possible for data used during the authentication process to be leaked to an outsider. Although the data is considered to be in the public domain, the secrecy property is desirable. This is because, by its very nature, biometric data cannot be replaced, changed or regenerated if it is stolen or compromised as it could be in other types of authentication such as with a password or smart card.

Biometric data can be revealed when received from the user through a biometric reader. It could also be leaked during data transmission, or disclosed by a corrupt machine involved in biometric matching.

These potential violations of privacy can be overcome by use of a TPM to verify system components such as the biometric reader or computing platform before the user's biometric data is received. See [23] for an example of this.

Since the biometric data can be captured, the hardware is required to be capable of ensuring that the biometric data has come from the user's live presentation, not, for example, a fake rubber finger. This is ensured by the liveness property.

The properties of the biometric authentication protocol should be proposed and stated clearly when creating a biometric authentication protocol.

7.2 Limitations and Future Work

We have studied various biometric authentication protocols and learnt about general issues associated with biometric authentication. This has enabled us to develop a different type of generic remote biometric authentication protocol. This thesis uses ProVerif to analyse the protocols. In addition, we have proposed properties that could be used as a basis for development of new protocols in the future. Even if ProVerif can automatically capture and manipulate messages across the channels, we have to create the forms of attacks to the protocol. Failure to identify attacks to the protocol could cause unsuccessful verification and analysis.

Next, we present potential future work in more detail. In chapter 4, we presented extension of the signature creation protocol. The protocol runs in sequence order. Therefore, the user has to perform biometric authentication before the signature is created each time. This is not practical if there are many documents that the user wishes to sign. To overcome this, the signature creation process should be separate from the biometric authentication process. However, there are security considerations associated with this: the user could sign a document that he has not agreed to sign. This is a risk because the document is shown to the user before biometric authentication is performed. Therefore, the user could be shown the desired document, but the card signs a different document.

In chapter 5, the 'Attestation-based remote biometric authentication' protocol [17] is verified and anlysed. The biometric data signed by the TPM is replayable. To fix this, the protocol simply generates a random number, a nonce, to be sent along with the biometric data. This number could be used to demonstrate re-use of the data from a previous login attempt; the recipient could figure out that the biometric data is not fresh.

Moreover, future work could build on the research in this thesis of remote biometric authentication for on-line banking. The proposed protocol requires the user to authenticate each time he wishes to perform a transfer transaction. It would be more practical if we could eliminate the redundant requirement for the user to present his biometric data each time. The solution could be to use a time stamp for each user's biometric verification. Nevertheless, the expiry time for the session must be considered.

Bibliography

[1] Salaiwarakul, A., Ryan, M.: Verification of Integrity and Secrecy Properties of a Biometric Authentication Protocol. Fourth Information Security Practice and Experience Conference (2008) 1-13.

[2] Salaiwarakul, A., Ryan, M.: Analysis of a Biometric Authentication Protocol for Signature Creation Application. Third International Workshop on Security (2008) 231-245.

[3] Ratha, N., Connell, J., Bolle, R.: Secure Data Hiding in Wavelet Compressed Fingerprint Images. Proceedings of the 2000 ACM Workshops on Multimedia (2000) 127-130.

[4] Ratha, N., Connell, J., Bolle, R.: A biometrics-based secure authentication system. Available at URL `http://www.research.ibm.com/ecvg/pubs/ratha-chall.pdf` (2008).

[5] Jain, A., Uludag, U.: Hiding Biometric Data. IEEE transactions on pattern analysis and machine intelligence (2003) 1494-1498.

[6] Khan, M., Zhang, J.: Implementing Templates Security in Remote Biometric Authentication System. International Conference on Computational Intelligence and Security (2006) 1396-1400.

[7] Waldmann, U., Scheuermann, D., Eckert, C.: Protected Transmission of Biometric User Authentication Data for Oncard-Matching. ACM Symposium on Applied Computing (2004) 425-430.

[8] Dolev, D. and Yao, A.C.: On the Security of Public Key Protocols. In Proceedings of 22nd IEEE Symposium on Foundations of Computer Science (1981) 350-357.

[9] Gobioff, H., Smith, S., Tygar, J. D., Yee, B.: Smart Cards in Hostile Environments. 2nd USENIX Workshop on Electronic Commerce (1996).

[10] Bond, M.: Chip and Pin (EMV) Point-of-Sale Terminal Interceptor. Available at URL `http://www.cl.cam.ac.uk/~mkb23/interceptor/` (2007).

[11] Blanchet, B.: ProVerif : Automatic Cryptographic Protocol Verifier User Manual (2005).

[12] Abadi, M., Fournet, C.: Mobile Values, New Names, and Secure Communication. POPL 2001.

[13] Abadi, M., Blanchet, B., and Fournet, C.: Just Fast Keying in the Pi Calculus. ACM Transactions on Information and System Security (TISSEC), 10(3):1-59, July 2007.

[14] Abadi, M., and Blanchet, B.: Computer-Assisted Verification of a Protocol for Certified Email. Science of Computer Programming, 58(1-2):3-27, October 2005. Special issue SAS'03.

[15] Kremer, S., Ryan, M.: Analysis of an Electronic Voting Protocol in the Applied Pi Calculus. In Proceedings of the European Symposium on Programming. Lecture Notes in Computer Science 3444. Springer Verlag (2005) 186-200.

[16] Delaune, S., Kremer, S., Ryan, M.: Coercion-resistance and Receipt-freeness in Electronic Voting. In 19th Computer Security Foundations Workshop. IEEE Computer Society Press (2006).

[17] Polon, T., Sander, S.: Attestation-Based Remote Biometric Authentication. Biometric Symposium: Special Session on Research at the Biometric Consortium Conference (2006) 1-5.

[18] Prabhakar,S., Paankanti,S., Jain, A.K.: Biometric Recognition: Security and Privacy Concerns. IEEE Security & Privacy (2003) 33-42.

[19] Chen, L., Person, S., Proundler, G., Chen, D., and Blanceheff, B.: How can you trust a computing platform? Proceedings of Information Security Solutions Europe Conference (ISSE 2000).

[20] Davida,G.I., Frankel, Y. and Matt, B.J.: On Enabling Secure Applications Through Off-line Biometric Identification. Security and Privacy. IEEE Symposium (1008) 148-157.

[21] Matsumoto, T., Matsumoto, H., Yamada, K., Hoshino, S.: Impact of Artificial Gummy Fingers on Fingerprint Systems Proceedings of SPIE Vol.4677. Optical Security and Counterfeit Deterrence Techniques IV (2002).

[22] Lowe, G.: An attack on the Needham-Schroeder public-key authentication protocol. Information Processing Letters 56 (1995) 131-133.

[23] Chen, L., Pearson, S., Vamvakas, A.: Trusted Biometric System. Available at URL `http://www.hpl.hp.com/techreports/2002/HPL-2002-185.pdf` (2002).

[24] Pearson. S.: How Can You Trust the Computer in Front of You?. Available at URL `http://www.hpl.hp.com/techreports/2002/HPL-2002-222.pdf` (2002).

[25] Abadi, M., Fournet, C.: Mobile values, new names, and secure communications. Proceedings of the 28th Annual ACM Symposium on Principles of Programming Languages (2001) 104-115.

[26] Blanchet, B.: An efficient cryptographic protocol verifier based on prolog rules. In Steve Schneider, editor, 14th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press (2001) 82-96.

[27] Blanchet, B.: Automatic Proof of Strong Secrecy for Security Protocols. In IEEE Symposium on Security and Privacy (2004) 86-100.

[28] Blanchet, B.: ProVerif Automatic Cryptographic Protocol Verifier User Manual (2005).

[29] Dolev, D. and Yao, A.C.: On the Security of Public Key Protocols. Proceedings of 22nd IEEE Symposium on Foundations of Computer Science (1981) 350-357.

[30] Pearson, S.: Trusted Computing Platforms, the Next Security Solution. Available at URL `http://www.hpl.hp.com/techreports/2002/HPL-2002-221.pdf` (2002).

[31] Drimer, S. and Murdoch, S.: Keep Your Enemies Close Distance Bounding Against Smartcard Relay Attacks. Available at URL `http://www.cl.cam.ac.uk/research/security/banking/relay/bounding.pdf` (2009).

[32] Polido, F.,et al: Common Biometric Exchange Format Framework. Available at URL `http://csrc.nist.gov/publications/nistir/NISTIR6529A.pdf` (2004).

[33] Cremers, C. and Lafourcade, P.: Comparing State Spaces in Automatic Security Protocol Verification. Proceedings of the 7th International Workshop on Automated Verification of Critical Systems (AVoCS'07) (2009) 49 - 63.

[34] The BioAPI Consortium.: BioAPI Specification, Version 1.1. Available at URL `http://www.bioapi.org/Downloads/BioAPI%201.1.doc` (2009).

[35] Ryan, M.: Introduction to the TPM 1.2. Available at URL http://www.cs.bham.ac.uk/~mdr/teaching/modules/security/intro-TPM.pdf (2009).

[36] Tusted Computing Group.: TPM Specification version 1.2 Parts 1-3. Available at URL http://www.trustedcomputinggroup.org/resources/ (2009).

[37] Chen, L., Ryan, M.: Attack, solution and verification for shared authorization data in TCG TPM. Available at URL http://www.markryan.eu/tpm/08-sharedAuthorisation.pdf (2009).

[38] Paulson, L,C.: The Inductive Approach to Verifying Cryptographic Protoocls. J.Computer Security 6 (1998).

[39] Cremers, C.: Scyther - Semantics and Verification of Security Protocols. Ph.D. dissertation, Eindhoven University of Technology (2006).

[40] Hussein, M., and Seret, D. : Extending TLS for trust delegation in home networks. Proceedings of IEEE International Conference on Advanced Communication Technology ICACT'06 (2006).

[41] Jurjens, J.: Code Analysis of a Biometric Authentication System Using Automated Theorem Provers. In 21st Annual Computer Security Application Conference (ACSAC 2005) (2005).

[42] Jurjens, J.: Sound Methods and Effective Tools for Model-based Security Engineering with UML. 27th International Conference on Software Engineering (2005).

[43] Lloyd, J., Jurjens, J.: Security Analysis of a Biometric Authentication System Using UMLsec and JML. 12th International Conference on Model Driven Engineering Languages and Systems Model'09 (2009).

[44] Lowe, G.: Casper A compiler for the analysis of security protocols. Journal of Computer Security (1998).

[45] Viti, C., Bistarelli,S.: Study and Development of a Remote Biometric Authentication Protocol. Technical Report IIT B4-04/2003 (2003).