Biometrics-Based Dynamic Authentication For Secure Services

Saif Mohammed Saeed Abdulla Al Aryani

Follow this and additional works at: https://scholarworks.uaeu.ac.ae/all_theses
Part of the Information Security Commons

Recommended Citation
https://scholarworks.uaeu.ac.ae/all_theses/456

This Thesis is brought to you for free and open access by the Electronic Theses and Dissertations at Scholarworks@UAEU. It has been accepted for inclusion in Theses by an authorized administrator of Scholarworks@UAEU. For more information, please contact faad.musa@uaeu.ac.ae.
United Arab Emirates University

College of Information Technology

Information Security Track

BIOMETRICS-BASED DYNAMIC AUTHENTICATION FOR SECURE SERVICES

Saif Mohammed Saeed Abdulla Al Aryani

This thesis is submitted in partial fulfillment of the requirements for the degree of Master of Science in Information Security

Under the Supervision of Dr. Zouheir Trabelsi

April 2016
I, Saif Mohammed Saeed Abdulla Al Aryani, the undersigned, a graduate student at the United Arab Emirates University (UAEU), and the author of this thesis entitled “Biometrics-based Dynamic Authentication for Secure Services”, hereby, solemnly declare that this thesis is my own original research work that has been done and prepared by me under the supervision of Dr. Zouheir Trabelsi, in the College of Information Technology at UAEU. This work has not previously been presented or published, or formed the basis for the award of any academic degree, diploma or a similar title at this or any other university. Any materials borrowed from other sources (whether published or unpublished) and relied upon or included in my thesis have been properly cited and acknowledged in accordance with appropriate academic conventions. I further declare that there is no potential conflict of interest with respect to the research, data collection, authorship, presentation and/or publication of this thesis.

Student’s Signature: [signature]
Date: 17/11/2016
Copyright © 2016 Saif Mohammed Saeed Abdulla Al Aryani
All Rights Reserved
Approval of the Master Thesis

This Master Thesis is approved by the following Examining Committee Members:

1) Advisor (Committee Chair): Zouheir Trabelsi
   Title: Associate Professor
   Information Security Track
   College of Information Technology
   Signature: [Signature]
   Date: 12/6/2016

2) Member: Yousef Al Hammadi
   Title: Assistant Professor
   Information Security Track
   College of Information Technology
   Signature: [Signature]
   Date: 12/6/2016

3) Member (External Examiner): Huwida Said
   Title: Associate Professor
   College of Innovation Technology
   Institution: Zayed University
   Signature: [Signature]
   Date: 1/6/2016
This Master Thesis is accepted by:

Dean of the College of Information Technology: Professor Omar El-Gayar

Signature ___________________ Date 15, 2016

Dean of the College of the Graduate Studies: Professor Nagi T. Wakim

Signature ___________________ Date 12, 2016

Copy 5 of 6
Abstract

This thesis proposes a secure authentication protocol against physical session hijacking attacks. In client/server technology, users establish sessions to access the services offered by the servers. However, using physical session hijacking attacks, malicious users may physically take control of ongoing sessions. Malicious users also can establish sessions with servers using stolen passwords. In both cases, the server will be communicating with the wrong user who pretends to be the real user.

The goal of this authentication protocol is to continuously and dynamically ensure that during an ongoing session the current session’s user is himself the real person that is known to the server. The proposed continuous and dynamic verification process is based on the use of the session user’s biometrics data. The proposed protocol uses the 40-byte Option field in the IP header to continuously and dynamically verify the session user’s biometrics. Since the biometrics data is potentially large in size, only random portion of biometrics data is used for authentication and is embed in the IP Option field. In this thesis, the focus is only on fingerprint and Iris biometrics data. The use of the IP Option field to embed the biometrics data will ensure that the proposed protocol is compatible with the current TCP/IP stack implementation. This would allow to not creating a new protocol or making major modification to the current TCP/IP stack implementation. This protocol has been simulated using Matlab to evaluate its performance. In addition, the authenticity and secrecy of the proposed protocol has been validated using Scyther tool.

Keywords: Physical session hacking attacks, Biometrics, IP header Option Field, Iris, Fingerprint, User authentication.
بروتوكول المصادقة الديناميكية للخدمات الأمنة المعتمد على القياسات الحيوية

المختصر

تقترح هذه الطريقة بروتوكول مصادقة أمن ضد هجمات الاختطاف المادية لجلسات الاتصال في تقنية الاتصال/المستخدم ، للمستخدمين إنشاء جلسات اتصال للوصول إلى الخدمات التي تقدمها الخوادم. ومع ذلك، قد يستطيع المستخدم الخبيث التحكم بجلسات الاتصال القائمة باستخدام هجمات الاختطاف المادية لهذه الجلسات. المستخدم الخبيث أيضا قد يستطيع إنشاء جلسة اتصال مع الخادم باستخدام كلمة مرور مسروقة. في كلتا الحالتين، الخادم سوف يتصل مع المستخدم الخاص الذي يتظاهر بأنه هو المستخدم الحقيقي.

هدف من هذا البروتوكول هو الحصول على مصادقة مستمرة و ديناميكية خلال جلسات الاتصال القائمة، ويؤكد بأن المستخدم الحالي هو فعلا المستخدم المعروف للخادم. العملية المستمرة و الديناميكية المقترحة لالتقاط تتأتى إلى البيانات الحيوية للمستخدم، والبروتوكول المقترح سوف يستخدم 40 بايت من المساحة المتوفرة في (IP option field) للتحقق بشكل مستمر و ديناميكية من البيانات IP الحيوية للمستخد. لأن البيانات الحيوية للمستخدم تكون كبيرة في الحجم عادة، فإن جزءًا بسيطة مختارًا بشكل عشوائي منها سوف يستخدم في عملية المصادقة و سوف يتم تضمينه في (IP (Option Field) في هذه الطريقة، سيتم التركيز على نوعين من البيانات الحيوية و هما بصورة البند و قرحية العين. إن استخدام المساحة المتاحة من تضمين البيانات الحيوية سوف يضمن أن البروتوكول المقترح سيكون موثوقاً لبروتوكول الإنترنت القائم الرسمي، هذا التضمين سوف يضمن عدم الحاجة لأي تطوير أو تعدل في مكدسات الإنترنت الحالية (TCP/IP stack). هذا البروتوكول تم محاكاة في هذه الأطراف باستخدام برنامج (Matlab) للتأكد من كفائته. بالإضافة إلى ذلك، فقد تم عملية التحقق من صحة و سرية البروتوكول باستخدام برنامج (Scyther).

مفهوم البحث الرئيسي: هجمات الاختطاف المادية لجلسات الاتصال، البيانات الحيوية، مجال IP option Field
Acknowledgements

I am especially thankful to Dr. Zouheir Trabelsi who introduced me to the exciting field of biometrics-based security and whose endless ideas and encouragement led to this and most other studies in which I have been involved.

I would like to thank my committee for their guidance, support, and assistance throughout my preparation of this thesis. I would like to thank the chair and all members of the Faculty of Information Technology at the United Arab Emirates University for assisting me all over my studies and research.

Special thanks go to my parents and wife who helped me along the way. I am sure they suspected it was endless. In addition, special thanks are extended to all my colleagues in information security program: Vasiqullah, Mohammed and Hussam.
Dedication

To my beloved parents, wife and children
# Table of Contents

Title .................................................................................................................................................. i 
Declaration of Original Work .............................................................................................................. ii 
Copyright ............................................................................................................................................ iii 
Approval of the Master Thesis ........................................................................................................... iv 
Abstract ................................................................................................................................................ vi 
Title and Abstract (in Arabic) ............................................................................................................ vii 
Acknowledgements ............................................................................................................................. viii 
Dedication ........................................................................................................................................... ix 
Table of Contents ............................................................................................................................... x 
List of Tables ......................................................................................................................................... x 
List of Figures ....................................................................................................................................... xii 
List of Abbreviations ........................................................................................................................... xv 
Chapter 1: Introduction ....................................................................................................................... 1 
Chapter 2: Background ....................................................................................................................... 4 
  2.1 User Authentication ..................................................................................................................... 4 
  2.2 Methodology of designing secure protocols .............................................................................. 6 
Chapter 3: Related Works .................................................................................................................. 9 
  3.1 Enhancing Security by Embedding Biometric Data in IP Header ........................................... 9 
  3.2 Biometrics Assisted Secure Network Transactions ................................................................. 10 
Chapter 4: Biometrics Theory .......................................................................................................... 12 
  4.1 Overview ..................................................................................................................................... 12 
  4.2 Fingerprint Minutiae Data .......................................................................................................... 13 
  4.3 ISO/IEC 19794-2:2005 Finger Minutiae Record Format ......................................................... 15 
  4.4 Iris Image Data ........................................................................................................................... 18 
  4.5 ISO 19794-6:2011 Iris Image Data Record Format ................................................................... 19 
Chapter 5: Proposed Biometrics-based Dynamic Authentication Protocol ..................................... 23 
  5.1 Overview ..................................................................................................................................... 23 
  5.4 Detailed Protocol Description .................................................................................................... 24 
  5.3 IP header’s Options Field Structure ........................................................................................... 27 
  5.4 Structure of challenge and response ............................................................................................ 31 
Chapter 6: Simulation and Results .................................................................................................... 39 
  6.1 Simulation Overview ..................................................................................................................... 39 
  6.2 Simulation Flow ............................................................................................................................ 43
6.3 Code Blocks ............................................................................................................. 44
6.4 Simulation Results ................................................................................................. 58
Chapter 7: Security Validation of Proposed Protocol .............................................. 61
  7.1 Scyther Tool Briefing ............................................................................................ 61
  7.2 The Security Validation ......................................................................................... 64
Chapter 8: Conclusion ............................................................................................... 67
Bibliography .................................................................................................................. 68
Appendix ....................................................................................................................... 71
  The header of single finger record .......................................................................... 71
  Iris Representation Header ....................................................................................... 72
List of Tables

Table 1: General Header Structure of Finger Minutiae Record................. 16
Table 2: Header of single finger structure.............................................. 16
Table 3: Code of finger position ............................................................. 17
Table 4: 5-byte finger minutia structure................................................ 18
Table 5: Iris general header structure...................................................... 19
Table 6: Structure of Iris image representation header............................. 20
Table 7: Detailed structure of fingerprint minutiae and Iris image challenges..... 33
Table 8: Detailed structure of single finger record header.......................... 71
Table 9: Detailed structure of Iris representation header............................ 72
List of Figures

Figure 1: UNIX password scheme .......................................................... 5
Figure 2: Smartcard hardware architecture ............................................. 6
Figure 3: Secure protocol triangle .......................................................... 6
Figure 4: IPBio packet flow .................................................................... 10
Figure 5: Protocol Description ............................................................... 11
Figure 6: Valleys and Ridges in Fingerprint ............................................. 13
Figure 7: Singular Points in Fingerprint (a) Core and (b) Delta ................. 14
Figure 8: Location and direction of ridge ending (encoded as valleys skeleton bifurcation point) .......................................................... 14
Figure 9: Location and direction of ridge bifurcation (encoded as ridge skeleton bifurcation point) ...................................................... 15
Figure 10: Location and direction of ridge ending skeleton point ............... 15
Figure 11: 5-byte finger minutia structure .............................................. 18
Figure 12: Uncropped Iris image or VGA Iris image ............................... 21
Figure 13: Cropped Iris image ............................................................... 21
Figure 14: Cropped and masked Iris images .......................................... 21
Figure 15: Image types and their approximate size .................................. 22
Figure 16: Flow chart of the proposed protocol (1 wrong response trial is allowed) .............................................................................. 24
Figure 17: Detailed description of proposed protocol ............................... 25
Figure 18: Structure of IP header field .................................................... 28
Figure 19: Structure of proposed Option field ........................................ 28
Figure 20: Structure of option type ......................................................... 28
Figure 21: Available functions of Option Type (RFC 791) ......................... 29
Figure 22: Structure of option length field ............................................. 30
Figure 23: Encryption of (challenge or response) ...................................... 30
Figure 24: Hash of (challenge or response) ............................................. 31
Figure 25: Challenge structure ............................................................... 32
Figure 26: ISO-19794-2:2005 finger position code ................................ 32
Figure 27: ISO-19794-6:2011 Eye label code .......................................... 32
Figure 28: Detailed structure of the proposed fingerprint minutiae challenge and its standard parts ................................................................ 34
Figure 29: Detailed structure of the proposed Iris image challenge and its standard parts .................................................................. 35
Figure 30: General structure of the proposed response ............................ 35
Figure 31: General structure of the proposed fingerprint minutiae response .... 36
Figure 32: Detailed structure of the proposed fingerprint minutiae response .... 36
Figure 33: General structure of the proposed Iris image response ............ 37
Figure 34: Detailed structure of the proposed Iris image response ............ 37
Figure 35: Simulation overview ............................................................. 39
Figure 36: Matlab console .................................................................... 40
Figure 37: Graphical User Interface of the simulation ........................................ 41
Figure 38: CSV file which represents the database structure ................................. 42
Figure 39: Simulation Flow .................................................................................. 43
Figure 40: Clientlogin structure ........................................................................... 45
Figure 41: Clientlogin Flow .................................................................................. 46
Figure 42: Server_auths_client flow ...................................................................... 47
Figure 43: Server_challenge flow .......................................................................... 48
Figure 44: Client_response flow ............................................................................ 50
Figure 45: Server_response flow .......................................................................... 52
Figure 46: Encrypt and decrypt flow ....................................................................... 53
Figure 47: ServerDB and ClientDB flow ................................................................. 55
Figure 48: AES algorithm ...................................................................................... 56
Figure 49: MD5 flow ............................................................................................. 57
Figure 50: Simulation results ................................................................................ 58
Figure 51: Runtime vs. Database size (in bytes) ..................................................... 59
Figure 52: Protocol minimum requirement (2 roles) .............................................. 62
Figure 53: Atomic items and events ...................................................................... 62
Figure 54: Set of possible security claims functions .............................................. 63
Figure 55: Scyther written code of the proposed protocol ..................................... 65
Figure 56: Security properties validation of the proposed protocol ....................... 66
### List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>CIK</td>
<td>Coordinate Information Key</td>
</tr>
<tr>
<td>CSV</td>
<td>Comma Separated Values</td>
</tr>
<tr>
<td>DB</td>
<td>Data base</td>
</tr>
<tr>
<td>DNA</td>
<td>DeoxyriboNucleic Acid</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>IEC</td>
<td>International Electrotechnical Commission</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>ISO</td>
<td>International Standards Organization</td>
</tr>
<tr>
<td>MD5</td>
<td>Message Digest 5 algorithm</td>
</tr>
<tr>
<td>PIN</td>
<td>Personal Identification Number</td>
</tr>
<tr>
<td>SPDL</td>
<td>Secure Protocol Description Language</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>TFTP</td>
<td>Trivial File Transfer Protocol</td>
</tr>
<tr>
<td>VGA</td>
<td>Video Graphic Adapter</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

Nowadays, secure services are greatly needed to protect institutions’ sensitive information. Secure services need to embody the requirements of information security, namely confidentiality (information is not disclosed to unauthorized person), integrity (accuracy and completeness of the transferred information), availability (information should serve its purpose and available when needed), authentication (ensuring that the communication agent is the one that it claims to be) and non-repudiation (proofing that the message is sent by the specified agent and received as well by the specified agent). The previous definitions are mentioned by (Stallings and Brown, 2012).

To implement security requirements, systems require secure protocols, such as Challenge and Response protocols which is explained by (Stallings, 2003). To achieve a strong security, these protocols use strong challenges. One of the best techniques to implement a stronger challenge is biometrics, which refer to the metrics related to human characteristics. Biometrics are measurable characteristics that are distinctive and can be used to label, describe or identify the individual. Most common biometrics that are used for identification are fingerprint, face, Iris, and DNA. The biometrics are represented in databases as binary values. The smallest part of biometrics data is known as Template Value. A given biometrics of a single person consists of a large number of templates.

This thesis main objective is to propose a secure protocol for continuously and dynamically authenticating both the user’s computer and the user himself that are involved in a given established session. The authentication of the computer is assured by normal cryptographic operations, such as symmetric cryptography and hash
functions. While the authentication of the user himself is achieved by continuously and dynamically verifying his/her biometrics data. Hence, the goal of this authentication protocol is to continuously and dynamically ensure that during an ongoing session the user himself is the person who is really using the device. In fact, a malicious user can either physically hijack an ongoing session after the authorized login takes place, or establish a session using stolen login and password. The continuous and dynamic verification of the user’ biometrics by the session’s server will allow to authenticate whether or not the current user for an ongoing session is the right person. If no, the session will be shut down by the server.

On the other hand, nowadays, most of the secure services run over IP networks. The proposed authentication protocol should be compatible with the different protocols running over IP networks, and should not require any modifications of the TCP/IP stack. To achieve this requirement, the proposed protocol uses the 40-byte Option field in the IP header to dynamically verify the session user’s biometrics. The use of the Option field to embed the biometrics data will ensure that the proposed protocol is compatible with any IP protocols. This would allow to not creating a new protocol or making major modification to the current TCP/IP stack implementation.

The identification of this protocol will be done in the Options field by identifying a unique number in the sub-field Option Type. This unique number in the Option Type will be used by the server and client of an ongoing session to identify that a given packet belongs to the proposed protocol. This protocol has the ability to authenticate users of different applications types, such as HTTP, FTP and TFTP.

The scope of this proposal is designing a protocol using pre-generated biometrics digital data for authentications in networks. The biometrics sensing devices
technology and the extraction methods of the biometrics data are out of this thesis’s scope. The determination of the strength or weakness of the used cryptographic algorithms is out of the scope of this thesis, as well. All used cryptographic algorithms are assumed to be perfect and secure. The type of encryption used on this thesis is symmetric cryptography. The key management of the secret is assumed to be done in earlier stage of the proposed solution.

The organization of the remaining part of the thesis is as following: Chapter-2 presents the background. Chapter-3 discusses related works. Chapter-4 covers the theory of biometrics and their standard digital representations. Chapter-5 explains the proposed protocol in details. Chapter-6 explains the conducted simulations and results. Chapter-7 evaluates the security of the proposed protocol using Scythe tool. Chapter-8 concludes the thesis.
Chapter 2: Background

2.1 User Authentication

User authentication is the process of verifying and identifying a user to the system based on claims that are defined by or for the system (Stallings and Brown, 2012). It consists of two steps, identification and verification. Verification is based on unique identifier of the user only. Identification is presenting and generating authentication information and compares it with the whole database that contains the enrolled data of individuals. An enrollment of the identifier should be done in earlier stage.

The authentication of a user is classified into four main categories:

- Something that identity knows (e.g. Password, PIN and Answers to challenge)
- Something that identity owns: (e.g. Smart Card and Physical Key)
- Something that identity is (static biometrics)
- Something that identity does (dynamic biometrics)

Password is widely used for user authentication. The system compares the user’s password with the enrolled password in the system database. In current operating systems, the password is stored as a hash of the original of the password with a salt value instead of depending on the password value itself. The UNIX password scheme is shown in Figure-1. The password is vulnerable to some attacks, such as dictionary attacks, electronic monitoring and popular password attacks (Stallings and Brown, 2012). Roles and policies are set to ensure having secure password.
Smart card is an example of authentication using user’s possessions. The hardware implementation of smart card is shown in Figure-2. There are three categories of smart card’s authentication protocols: static, dynamic password generator and challenge-response. In static, the user authenticates himself to the token, and then the token authenticate the user to the system. In dynamic password generator, the token generates a unique password dynamically, and the password is entered to the computer either manually by the user or electronically via the token. In the challenge response authentication protocol, the system generates a challenge and smart card response based on that challenge (e.g. the challenge is encrypted by public key and the token decrypts it using private key).
Biometrics is the science where the person is identified or verified using the metrics of his biological characteristics. Biometrics are explained in details in chapter-4 of this thesis.

2.2 Methodology of designing secure protocols

Designing a secure protocol is a trade-off between three factors: security, efficiency and scalability. The triangle of secure protocol design is shown in Figure-3.

Figure 2: Smartcard hardware architecture

Figure 3: Secure protocol triangle
The security of the protocol is achieved by the following known information security requirements:

Confidentiality: (e.g. confidentiality of the transmitted data is achieved by symmetric encryption)

Integrity: (e.g. integrity of transmitted data is achieved by message authentication code)

Availability: information should serve its purpose and available when needed

Non-repudiation: (e.g. sending user’s biometrics will ensure non-repudiation)

Authentication: (e.g. user is verified by his biometrics data)

The efficiency of a secure protocol is a term used to describe the properties related to the computational resources. The designer of any protocol should optimize the usage of these computational resources. The efficiency is measured by some criteria, such as:

Throughput: (e.g. the transmitted data rate per second).

Memory usage: (e.g. The required memory to perform the protocol)

Power consumption: (e.g. the amount of power consumption of each challenge-response algorithm)

Cost : (e.g. the cost of computers, routers, software to implement the protocol)

For example, implementing 256-bit encryption will provide more security than 128-bit if the same algorithm is used. However, in the other hand, the memory usage and power consumption will increase as well.
The scalability is the ability of the protocol to handle the growing amount of work. The designer of any protocol should expect that his protocol will be able to interact with the rapid improvement of security requirements. The scalability is measured by some criteria, such as:

Compatibility: (e.g. the protocol is compatible with different operating systems, and the protocol is compatible with any TCP/IP protocol)

Upgradeability: (e.g. the protocol can be upgraded to support any new type of biometrics authentication that is not supported now).

For example, supporting different operating systems will increase the scalability of the system, but it will increase the surface of the system which will encourage the attackers to find vulnerabilities in the system.
Chapter 3: Related Works

3.1 Enhancing Security by Embedding Biometric Data in IP Header

(Lee, Kim and Yoo, 2007) introduces an IPBio theory. They proposed a solution to have a continuous authentication of all packets coming from the live sender (client). They proposed to include a portion of the biometrics data of the client for per-packet authentication to digest the IP datagram. Their solution is started by initial exchange of the session key that is used to encrypt the initial biometric data. They depend on Lamport’s technique to ensure a secure way of authenticating the client. After confirmation by the receiver (server), the session key will not be used and a random portion of the bio-data is used to encrypt the digested value of the packet payload. A new protocol was created by them (IPPROTO_BIO_KEY) for login packet session. This requires a modification in the server TCP/IP stack.

After establishing the session, the user opens a TCP connection to the server by sending (SYN packet) containing his biometrics data (collected at run time) encrypted with the session key. The server decrypts the bio-data and verifies the data by comparing it with the pre-registered data in server database. The server then, saves the current biometric data for this session. An Acknowledgement SYN is sent by the server if the verification is successful. This Acknowledgement SYN differs from the normal Acknowledgement SYN because it contains a direction about the required portion of the biometric data. The client responds with the final ACK, to complete 3-way handshake. The response contains a portion of bio-data based on the server direction. The packet format of SYN and ACK is explained in Figure-4. The portion of bio-data will be used to digest all packets sent by the client during the session.
3.2 Biometrics Assisted Secure Network Transactions

(Kunnill, Pillai and Milshtein, 2007) proposed an authentication protocol based on fingerprint image to generate an authentication key instead of depending on Password or PINS. The authentication process was done by selecting random areas from the fingerprint image as a key. An additional layer which is blood vessel map is used to determine that a live finger is being used.

They propose a trusted third party based approach for providing keys and timestamps for information exchange as shown in Figure-5. In initial phases; the biometric reader sends a message to token server and remote server before acquiring fingerprint data. Then, the remote server generates a CIK (Coordinate Information Key) which contains index of fingerprint portion stored in the remote server as a key to the token server. This key is time-stamped by token server and forwarded to the biometric reader and remote server. Once key is received, the biometrics reader acquires the fingerprint image and blood vessel. After that, the reader sends the
authentication key along with blood vessel to remote server. Biometrics reader also forwards the time stamp and CIK to the remote server. The remote server verifies the timestamp along with CIK and performs matching of the blood vessel map and the authentication key by calculating the authentication key locally.

Figure 5: Protocol Description
Chapter 4: Biometrics Theory

4.1 Overview

Biometrics is the science where the person is identified or verified using the metrics of his biological characteristics. The biological characteristics are unique for each person. The biometrics is classified into two parts: static biometrics and dynamic biometrics. The static biometrics are the things that (person is), such as Fingerprint, IRIS, Retina, Face, and DNA. The dynamic biometrics are the things that (person does), such as writing style, voice level and typing rhythm. The focus in this thesis is only on the static biometrics of the individual for authenticating the network parties to each others. The advantages of using bio-authentication are:

- It is Unique for every individual and cannot be lost or stolen by another individual.
- It is difficult to transfer or copy the biometrics data.
- Enhancing the security of the authentication.
- Solving repudiation issues.
- Provide both confidence and convenience of the user.

The bio-authentication has some disadvantages, such as:

- Biometrics are static features which cannot be “reset” such as passwords.
- The bio-data comparison is Matching Score not (Yes/No like passwords) due to the nature of sensing technologies.
- The privacy is big concern where not all of people accept.
In this thesis, the focus only on two types of biometrics which are Fingerprint minutiae data and Iris Image Data. They are explained based on ISO/IEC-19794 standard. The scope of this thesis is to use the digital stored data of biometrics in the proposed authentication protocol. The mechanism of biometrics sensing technologies, features extraction and mathematics of analyzing are out of the scope. The generation of the digital representation can be done by two methods: Feature-based method or Image-based method. Feature-based method extracts the important features of the biometrics image and set them in systematic way. Image-based method directly uses the image for matching without extraction any feature.

4.2 Fingerprint Minutiae Data

The fingerprint is the pattern of interleaved ridges and valleys on the tip of the finger as shown in Figure-6. The method used to extract this type of biometrics is feature-based method. The features of any fingerprint are mainly classified into two main categories: Level-1 Features and Level-2 Features.

![Valleys and Ridges in Fingerprint](image)

Figure 6: Valleys and Ridges in Fingerprint

Level-1 mainly focuses on finding the general behavior of the fingerprint in terms of Singular Points. The singular points are places where the ridges orientations changes abruptly. There are two types of singular points: core and delta s shown in Figure-7. Core is a place where group of ridges comes and change their directions
with the same angle. A delta is a place where three ridges are seem to meet. The fingerprint usually has (0, 1 or 2) cores and (0, 1, or 2) deltas.

Figure 7: Singular Points in Fingerprint .(a) Core and (b) Delta

Level-2 features mainly focuses on defining the fingerprint minutiae locations in the fingerprint image. The locations where (ridge or valley) ends, splits or merges with another (ridge or valley) are termed as fingerprint minutiae. As shown in Figures-(8, 9 and 10), there are three types of minutiae which are: Valley Skeleton Bifurcation Point, Ridge Skeleton Bifurcation Point and Ridge Skeleton End Point.

Figure 8: Location and direcrion of ridge ending (encoded as valleys skeleton bifurcation point)
Figure 9: Location and direction of ridge bifurcation (encoded as ridge skeleton bifurcation point)

Figure 10: Location and direction of ridge ending skeleton point

4.3 ISO/IEC 19794-2:2005 Finger Minutiae Record Format

This part represents the standard digital representation of fingerprint based on features-based method. The record is organized into two parts:

General Header: Fixed length 14-bytes

Single Finger Record which consists of:

- 32-bytes header of single finger
- Series of (5 or 6) bytes minutiae descriptions.
- Extended data which may contain extra information about the fingerprint, such as singular points.

The General Header
The general header of Finger Minutia Record consists of several parts as shown in Table-1.

Table 1: General Header Structure of Finger Minutiae Record

<table>
<thead>
<tr>
<th>Name of Field</th>
<th>Size</th>
<th>Valid Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format Identifier</td>
<td>4 bytes</td>
<td>464D5200 Hex</td>
<td>ASCII (FMR) = Finger Minutiae Record followed by (00 Hex)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(<code>F</code> ‘M’ ‘R’ 00 Hex)</td>
<td></td>
</tr>
<tr>
<td>Version of the standard</td>
<td>4 bytes</td>
<td>30333000 Hex</td>
<td>Standard Version = ASCII(030) followed by (00 Hex)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(`0’ ‘3’ ‘0’ 00 Hex)</td>
<td></td>
</tr>
<tr>
<td>Length of total record in bytes</td>
<td>4 bytes</td>
<td>24 Hex to FFFFFFFF Hex</td>
<td>Entire Length of Finger Minutiae Data Record in bytes</td>
</tr>
<tr>
<td>No. of finger views</td>
<td>2 bytes</td>
<td>0001 Hex to 00B0 Hex</td>
<td>1 to (10 fingers+1 unknown) times 16 = 176</td>
</tr>
</tbody>
</table>

The Header of Single Finger

The length of each finger header is 32 bytes. The important parts only are represented in the following Table-2. The whole explanation is given in Table-9 (appendix).

Table 2: Header of single finger structure

<table>
<thead>
<tr>
<th>Name of Field</th>
<th>Size</th>
<th>Valid Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finger position</td>
<td>1 bytes</td>
<td>0 to 10</td>
<td>Details in Table-3</td>
</tr>
<tr>
<td>Minutiae Field Length</td>
<td>4 bits</td>
<td>5 or 6</td>
<td>Number of bytes required to represent each minutia (5 bytes in this thesis)</td>
</tr>
</tbody>
</table>
Ridge Ending Type | 4 bits | 0 or 1 |
|------------------|--------|-------|

Determine the location of ridge ending shall be recorded:
- 0: Minutia indicated using three valley bifurcations.
- 1: Minutia indicated using ridge skeleton endpoints

Number of Minutiae | 1 byte | Up to 256 |
|-------------------|--------|-----------|

Number of Minutiae extracted and encoded for this fingerprint

Remaining parts | 29 bytes |
|----------------|----------|

Details in Table-9 (Appendix)

Table 3: Code of finger position

<table>
<thead>
<tr>
<th>Finger Position</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unknown Finger</td>
<td>0</td>
</tr>
<tr>
<td>Right Thumb</td>
<td>1</td>
</tr>
<tr>
<td>Right Index Finger</td>
<td>2</td>
</tr>
<tr>
<td>Right Middle Finger</td>
<td>3</td>
</tr>
<tr>
<td>Right Ring Finger</td>
<td>4</td>
</tr>
<tr>
<td>Right Little Finger</td>
<td>5</td>
</tr>
<tr>
<td>Left Thumb</td>
<td>6</td>
</tr>
<tr>
<td>Left Index Finger</td>
<td>7</td>
</tr>
<tr>
<td>Left Middle Finger</td>
<td>8</td>
</tr>
<tr>
<td>Left Ring Finger</td>
<td>9</td>
</tr>
<tr>
<td>Left Little Finger</td>
<td>10</td>
</tr>
</tbody>
</table>

Series of 5 bytes minutiae descriptions (Metric Description)
In this part, the minutiae are encoded and the structure of each minutia is shown in Figure-11. The detailed description of finger minutia is explained in Table-4.

![5 byte finger minutia structure](image)

**Figure 11: 5-byte finger minutia structure**

<table>
<thead>
<tr>
<th>Name of Field</th>
<th>Size</th>
<th>Valid Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>2 bits</td>
<td>0, 1 or 2 (3 reserved)</td>
<td>• 0: Other Type&lt;br&gt;• 1: Ridge Ending&lt;br&gt;• 2: Ridge Bifurcation</td>
</tr>
<tr>
<td>X-Coordinate</td>
<td>14 bits</td>
<td>Up to 16384</td>
<td>Indicates the X-coordination of the pixel contains the Minutia</td>
</tr>
<tr>
<td>Reserved</td>
<td>2 bits</td>
<td>Reserved and set to 0</td>
<td>For future use</td>
</tr>
<tr>
<td>Y-Coordinate</td>
<td>14 bits</td>
<td>Up to 16384</td>
<td>Indicates the Y-coordination of the pixel contains the Minutia</td>
</tr>
<tr>
<td>Angle</td>
<td>8 bits</td>
<td>Up to 255</td>
<td>• Represents the angle of the minutia&lt;br&gt;• Angle is recorded in unit of ( \frac{1.40625}{360} ) (360/256)</td>
</tr>
</tbody>
</table>

**4.4 Iris Image Data**

The Iris is the part of the eye which centers the eye and surrounds the pupil and controls its size. It is located behind the cornea and in front of the lens. The method used to extract this type of biometrics is Image-based method. The Iris differs between individuals in terms of size, color, round center, pupil size and other factors.
4.5 ISO 19794-6:2011 Iris Image Data Record Format

This part represents the standard digital representation of Iris based on image-based method. The record is organized into two parts:

General Header: Fixed length 16-bytes

Iris Image:

- Iris Representation Header: 53 bytes
- Image Representation Body: series of pixels (8 bits)

The General Header

The general header of Iris Image Record consists of several parts, as shown in Table- 5

Table 5: Iris general header structure

<table>
<thead>
<tr>
<th>Name of Field</th>
<th>Size</th>
<th>Valid Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format Identifier</td>
<td>4 bytes</td>
<td>49495200 Hex ('I' 'I' 'R' 00 Hex)</td>
<td>IIR = Iris Image Record followed by (00 Hex)</td>
</tr>
<tr>
<td>Version Number</td>
<td>4 bytes</td>
<td>30323000 Hex ('0' '2' '0' 00 Hex)</td>
<td>Standard Version = ASCII(020) followed by (00 Hex)</td>
</tr>
<tr>
<td>Length of record</td>
<td>4 bytes</td>
<td>69 Hex to FFFFFFFF Hex</td>
<td>Entire Length of Iris Image Data in bytes</td>
</tr>
<tr>
<td>No. of Iris Representations</td>
<td>2 bytes</td>
<td>1 to 65535</td>
<td>At least 1 representation required</td>
</tr>
<tr>
<td>Certification Flag</td>
<td>1 byte</td>
<td>00Hex</td>
<td>No certification flag for ISO/IEC-19749</td>
</tr>
</tbody>
</table>
**Number of Eyes Represented**

<table>
<thead>
<tr>
<th>Name of Field</th>
<th>Size</th>
<th>Valid Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eye Label</td>
<td>1 byte</td>
<td>0,1 or 2</td>
<td>• 0: Undefined Eye&lt;br&gt;• 1: Right Eye&lt;br&gt;• 2: Left Eye</td>
</tr>
<tr>
<td>Image Type</td>
<td>1 bits</td>
<td>1, 2, 3 or 7</td>
<td>• 1: Un-cropped Iris Image (see Figure-12)&lt;br&gt;• 2: VGA Iris Image (see Figure-12)&lt;br&gt;• 3: Cropped Iris Image (see Figure-13)&lt;br&gt;• 7: Cropped and masked Iris Image (see Figure-14)</td>
</tr>
<tr>
<td>Image Format</td>
<td>1 byte</td>
<td>2, 10, or 14</td>
<td>• 2: MONO_RAW&lt;br&gt;• 10: MONO_JPEG2000&lt;br&gt;• 14: MONO_PNG</td>
</tr>
<tr>
<td>Remaining parts</td>
<td>29 bytes</td>
<td></td>
<td>Details in Table-10 (appendix)</td>
</tr>
</tbody>
</table>

---

**The Iris Representation Header**

The length of Iris representation header is 53 bytes. The important parts only are represented in the following Table-6. The whole explanation is given in Table- at appendix.
This field contains the data of the captured Iris Image. Each pixel of uncompressed gray-scale is occupying 8 bits (256 gray levels). If the Image is compressed, the pixel should be compressed using the same technique. The approximate record size of each type of Iris images is represented in Figure-15 (source: ISO-19794-6:2011).
<table>
<thead>
<tr>
<th>Role</th>
<th>Recommended Type and Compressor</th>
<th>Target Record Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>IMAGE_TYPE_UNCROPPED PNG lossless or JPEG2000 lossless</td>
<td>2kB 4kB 8kB 16kB 32kB 64kB 128kB 256kB</td>
</tr>
<tr>
<td>All</td>
<td>IMAGE_TYPE_VGA (640x480) PNG lossless or JPEG2000 lossless</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>IMAGE_TYPE_CROPPED PNG lossless or JPEG2000 lossless</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>IMAGE_TYPE_CROPPED_AND_MASKED PNG lossless or JPEG2000 lossless</td>
<td></td>
</tr>
<tr>
<td>1:N</td>
<td>IMAGE_TYPE_CROPPED JPEG2000</td>
<td></td>
</tr>
<tr>
<td>1:N</td>
<td>IMAGE_TYPE_CROPPED_AND_MASKED JPEG2000</td>
<td></td>
</tr>
<tr>
<td>1:1</td>
<td>IMAGE_TYPE_CROPPED JPEG2000</td>
<td></td>
</tr>
<tr>
<td>1:1</td>
<td>IMAGE_TYPE_CROPPED_AND_MASKED JPEG2000</td>
<td></td>
</tr>
</tbody>
</table>

Figure 15: Image types and their approximate size
Chapter 5: Proposed Biometrics-based Dynamic Authentication Protocol

5.1 Overview

The main objective of the proposed protocol is to authenticate session clients in a continuous and dynamic manner using biometrics data. The protocol is compatible with the TCP/IP protocols. It is important to mention that the biometrics data is potentially large in size which can affect the performance of the communication. Hence, only random part of the biometrics data will be embedded in the network exchanged packets. This solution will introduce a unique combination between the client’s device IP address and the client’s biometrics data. The proposed biometric-based IP protocol is called Bio-IP. Practically, the random biometrics portion is embedded in the Option field of the IP header. Embedding the biometrics in the Options field will ensure both scalability and compatibility. The system will be compatible with any system since it is not proposing a new protocol that can be dropped by secure routers or operating systems. The client’s biometrics data is stored in the server by the Server Administrator at an early stage.

The proposed solution uses a number of steps. The flow chart of the proposed protocol is shown in Figure-16. After logging to the server using an ID and password, the server dynamically (using random times) challenges the user to send a random portion of his biometrics data for authentication. In case of wrong response, the server will again challenge the client (e.g., two wrong trials are allowed). If the user’s responses to all challenge requests are wrong, then the server will shutdown the session. Practically, once a response from the client is received, the server will match the embedded biometrics data with the stored one. Since the biometrics sensing technology will not give always an exact matching to the stored biometrics in
the server, the matching algorithm used by the proposed protocol is not based on an exact matching. In fact, it uses a score matching algorithm based on an accepted threshold, which depends on some comparison approaches. Hamming distance is an example of matching score algorithm (Li and Jain, 2009).

Figure 16: Flow chart of the proposed protocol (1 wrong response trial is allowed)

5.4 Detailed Protocol Description

The proposed protocol uses 11 steps as shown in Figure-17. The first 3 steps are for user login using user’s ID and password. The remaining 8 steps are related to the biometrics authentication process, during which the server attempts to issue
challenge requests to the client for continuous and dynamic authentication. After completing step-11, the server will send another challenge after random time to ensure continuous and dynamic authentication during the session period. Using a predefined random time interval, the server will keep sending challenge requests to continuously and dynamically ensure client authentication during the whole session period.

Figure 17: Detailed description of proposed protocol
The detailed description of the protocol steps are discussed in detail as following:

**Step-1**
- The user tries to login to the server using his user ID and Password

**Step-2**
- The server verifies the login by comparing:
  - User ID =? User ID
  - Hash(Password) =? Hash (Password)

**Step-3**
- The server notifies the user on the login attempt results.

**Step-4**
- The server generates a random challenge (C) after random time
- Server encrypts the challenge for confidentiality Enc(C)
- Server hashes the challenge for integrity Hash (C)
- Both encrypted and hashed challenge values are embedded in IP Option filed.

**Step-5**
- The server sends a packets to the client <IP SOURCE, IP DESTINATION, Enc(C), Hash(C)>

**Step-6**
- The user verifies the received data from server to ensure data integrity:
  - Hash (Dec(Enc(C))=? Hash(C)

**Step-7**
- The user generates the response (R) which contains the value of bio-template.

- The user applies XOR operation for challenge and response (C XOR R)

- User encrypts the XOR value for confidentiality Enc(C XOR R)

- Server hashes the XOR value for integrity Hash(C XOR R)
• Both encrypted and hashed XOR values are embedded in IP Option filed.

**Step-8**

• The client sends a packets to server \(<\text{IP}_{\text{SOURCE}}-, \text{IP}_{\text{DESTINATION}}, \text{Enc(C XOR R)}, \\text{Hash(C XOR R)}>\)

**Step-9**

• The server verifies the received data from user to ensure data integrity:
  
  – Hash (Dec(Enc(C XOR R))) =? Hash(C XOR R)

**Step-10**

• The server verifies the bio-template that extracted from the response.

• A matching score algorithm is used for comparison (With accepted threshold).

  Different mechanisms can be applied for matching score, such as Hamming Distance

**Step-11**

• The server notifies the user if the login (Accept, Another trial or End the session).

• Three trials are given to the client before server shuts down the session.

**Step-12**

In case of successful bio-authentication; after random time the server generates another challenge to the user. It will go back to step-4 and complete rest of steps to ensure dynamic authentication of the user.

---

### 5.3 IP header’s Options Field Structure

(Internet standards RFC 791, 1981) stated that the size of the Option field is maximum equals to 40 bytes (10 words), while the proposed protocol requires only 36 bytes (9 words). The structure of the IP header is shown in Figure-18.
The proposed structure of the IP option field is divided into five parts as shown in Figure-19.

Option Type

The standard length of Option type in Options field is 1 byte (8 bits) as shown in Figure-20.
Copy Flag is set to 1 if the Option is intended to be copied to all fragments if the data is fragmented, and it is set to 0 if the option should not copied. In this protocol, copy flag is set to 1 to avoid fragmentations problems.

Option class indicates four potential values that explain the general category into Option which belongs. Two used values, 0 for control option and 2 for debugging and measurements. The values (1 and 3) are reserved for future use. Option Number specifies the kind of option among 32 potential values.

Option type can be equal to some values that are indications to do some standard functions. Theses function and their corresponding values are shown in Figure-21. In the proposed protocol, a different value is chosen as identification of the proposed protocol for both server and client instead of creating a new protocol number in the protocol field of IP header.

<table>
<thead>
<tr>
<th>Copy</th>
<th>Class</th>
<th>Number</th>
<th>Value</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>EOF - End of Options List</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>NOP - No Operation</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>7</td>
<td>7</td>
<td>RR - Record Route</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td>ZSU - Experimental Measurement</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>11</td>
<td>11</td>
<td>MTUP - MTU Probe</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>12</td>
<td>12</td>
<td>MTUR - MTU Reply</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>15</td>
<td>15</td>
<td>ENCODE - ??</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>25</td>
<td>25</td>
<td>QS - Quick-Start</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>30</td>
<td>30</td>
<td>EXP - RFC3692-style Experiment</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>4</td>
<td>68</td>
<td>TS - Time Stamp</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>15</td>
<td>82</td>
<td>TR - Traceroute</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>30</td>
<td>94</td>
<td>EXP - RFC3692-style Experiment</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td>130</td>
<td>SEC - Security</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
<td>131</td>
<td>LSR - Loose Source Route</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>5</td>
<td>133</td>
<td>E-SEC - Extended Security</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>6</td>
<td>134</td>
<td>CIPS0 - Commercial Security</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>8</td>
<td>136</td>
<td>SID - Stream ID</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>9</td>
<td>137</td>
<td>SSR - Strict Source Route</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>14</td>
<td>142</td>
<td>VISA - Experimental Access Control</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>16</td>
<td>144</td>
<td>IMHID - IMI Traffic Descriptor</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>17</td>
<td>145</td>
<td>EIP - Extended Internet Protocol</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>19</td>
<td>147</td>
<td>ADDEXT - Address Extension</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>20</td>
<td>148</td>
<td>RTRALT - Router Alert</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>21</td>
<td>149</td>
<td>- Unassigned (Released 18 October 2005)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>22</td>
<td>150</td>
<td>DPS - Dynamic Packet State</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>23</td>
<td>151</td>
<td>UMP - Upstream Multicast Pkt.</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>24</td>
<td>152</td>
<td>EXP - RFC3692-style Experiment</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>30</td>
<td>158</td>
<td>FNN - Experimental Flow Control</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>13</td>
<td>205</td>
<td>EXP - RFC3692-style Experiment</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>30</td>
<td>222</td>
<td>EXP - RFC3692-style Experiment</td>
</tr>
</tbody>
</table>

Figure 21: Available functions of Option Type (RFC 791)
Option Length

Option length indicates the full length of the IP header Option field in Bytes as shown in Figure-22. Maximum value is 40 bytes which is (00101000). In the proposed protocol, the length is 36 bytes which is (00100100).

Figure 22: Structure of option length field

Encrypted Challenge

This part contains the encryption of challenge or response. The length of this part is 16 bytes (128 bits) as shown in Figure-23. Symmetric encryption with 128 bit block is used. AES-128 is the recommended encryption algorithm. Any AES candidate algorithms, such as (Serpent, Twofish, RC6 or MARS) with 128 bit block size can be used as alternative encryption algorithm.

Figure 23: Encryption of (challenge or response)
Hashed Challenge

This part contains the hashed value of challenge or response. The length of this part is 16 bytes (128 bits) as shown in Figure-24. MD5 is proposed algorithm since its length is 128 bit. SHAKE-128 can be used as alternative hash algorithm.

![Figure 24: Hash of (challenge or response)](image)

Padding

The length of this part is two bytes. It is set to zero values. This part can be used for any future enhancement of the proposed protocol.

5.4 Structure of challenge and response

The challenge and response field are structured according to the type of the biometrics used (e.g. Fingerprint or IRIS).

Challenge Structure

The nature of the challenge, is requesting a portion of the client’s biometrics data. The challenge consists of other fields, such as the type of the requested biometrics data (fingerprint or Iris), the User ID, and some detailed extracted from the standard header of the requested biometrics data. The server sends only the start index of the
requested biometrics portion. The challenge structure is divided into five parts as shown in Figure- 25.

![Challenge structure](image)

Figure 25: Challenge structure

![ISO-19794-2:2005 finger position code](image)

Figure 26: ISO-19794-2:2005 finger position code

![ISO-19794-6:2011 Eye label code](image)

Figure 27: ISO-19794-6:2011 Eye label code
The detailed description of biometrics challenge part is shown Table-7.

Table 7: Detailed structure of fingerprint minutiae and Iris image challenges

<table>
<thead>
<tr>
<th>Index</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0:15]</td>
<td>ID of user</td>
<td>More than 65.5K possible values</td>
</tr>
<tr>
<td>[16:47]</td>
<td>Bio Type</td>
<td>These values are depending on ISO standard.</td>
</tr>
<tr>
<td></td>
<td>0x464D5200 - Fingerprint</td>
<td>It Consists of four ASCII Letters</td>
</tr>
<tr>
<td></td>
<td>0x49495200 – Iris</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Others – Reserved</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>These values are depending on ISO standard.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>It represents: (which finger of the user, or which eye) is requested for authentication</td>
</tr>
<tr>
<td>[48:55]</td>
<td>If fingerprint bio-type:</td>
<td>These values are depending on ISO standard.</td>
</tr>
<tr>
<td></td>
<td>0x00-0x0A – Finger position</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Others – Reserved</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x00,0x01,0x02 – Eye label</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Others – Reserved</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>[56:63]</td>
<td>If fingerprint bio-type:</td>
<td>These values are depending on ISO standard.</td>
</tr>
<tr>
<td></td>
<td>Padding (All zeros)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x01,0x02,0x07,0x07</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Others</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>[64:95]</td>
<td>Bio-Header specification padding</td>
<td>It can be used for future enhancement of the protocol</td>
</tr>
</tbody>
</table>
The detailed structure of the standard parts of fingerprint minutia challenge is shown in Figure-28.

![Proposed Fingerprint Challenge Structure](image)

Figure 28: Detailed structure of the proposed fingerprint minutiae challenge and its standard parts

The detailed structure of the standard parts of Iris image challenge is shown in Figure-29.
Response Structure

The nature of response is sending the required portion of biometrics data of fingerprint minutiae or Iris pixels. The response value is called the biometrics template as shown in Figure-30.
For the fingerprint, the template value is consists of:

Three Sequenced fingerprint minutiae values (5 bytes each) starting from the requested index

Number of cores in the fingerprint (4 bits)

Number of deltas in the fingerprint (4 bits).

Figure 31: General structure of the proposed fingerprint minutiae response

Figure 32: Detailed structure of the proposed fingerprint minutiae response
For the IRIS; the template value is consists of:

Sequence of 15 Iris pixels values (8 bits each) starting from the requested index

Figure 33: General structure of the proposed Iris image response

Figure 34: Detailed structure of the proposed Iris image response

The proposed protocol states that if the requested template or any field of the challenge is out of range, the response sets the corresponding field to zero
Chapter 6: Simulation and Results

6.1 Simulation Overview

This simulation is implemented in MatLab environment. Databases are structured in simple comma delimited CSV files but contain all the necessary information to prove and show the feasibility of IP layer biometric authentication.

Figure 35: Simulation overview

Figure -35 shows the simulation overview. The user runs the simulation code blocks using the Matlab console as shown in Figure-36. The simulation code blocks
instantiates the graphical user interface as shown in Figure-37, where the user can provide the login information and continues with the simulation by pressing the LOGIN push button. The simulation code blocks reads and parses the client and server databases. The CSV database file is shown in Figure-38. While the simulation is running, output messages are fed back to the graphical user interface as well as a more detailed one in the MatLab command line interface. Simulation variables can also be observed on the MatLab base workspace as all simulation code blocks reads and updates these variables throughout the simulation. The user can do again the biometric part of the authentication protocol by pressing RAND AUTH push button. In the simulation, the client is allowed to provide the asked bio-template only three times, otherwise, the simulation terminates. The bio-template is accepted or not based on their matching score. The matching score is calculated using Hamming distance algorithm.
Figure 37: Graphical User Interface of the simulation
Figure 38: CSV file which represents the database structure
6.2 Simulation Flow

Figure 39: Simulation Flow

Figure 39 shows the simulation flow and the major code blocks. The code blocks are called in a chained manner similar to the sequential events of biometrics-based dynamic authentication for secure services.

The simulation starts by calling the init function. Init initializes the simulation. Then, it calls the clientlogin function. Clientlogin function creates the graphical interface. When the user clicks the login push button, all the text box information is extracted by clientlogin function and are compared with the local client database. It verifies login information to the server database by calling server_auths_client.
function. If everything matches in the server database, then it calls the server_challenge function to start biometric part of the authentication protocol. In server_challenge, the server generates a challenge packet which contains the type of biometric information asked (Iris or fingerprint), and the asked index. Server_challenge also calculates the MD5 checksum, encrypt the packet, and send it over to client. This is done by calling the client_response function. In client response function the encrypted server challenge is decrypted, then the hash is calculated and compared with the received hash. If the two hashes matches, this means that the server challenge packet is not modified and valid. After this, the client pulls out the biometric template of index and type asked by the server. The client applies an XOR operation of both challenge and response. Then, the client respond with the encrypted XOR value together with its corresponding MD5 hash. Calling server_response function corresponds to the receiving the client response packet and MD5 hash. The server decrypts this, calculate the hash, and compare the calculated hash to the received hash. Then the server sends a result message to the client ending the IP layer biometric authentication protocol.

6.3 Code Blocks

Block-1: Client Login

The function of this code block is summarized as follows:

Contains the functions that creates the graphical interface

Defines the login pushbutton call back function

Initialize all the client variables
Get the user input from the graphical user interface (user ID, password, IP address)

Generates the user login packet from the user input

Loads the local client database by calling clientdb function

Cross checks the local client database if the login ID and password are correct

If the login information does not exist in the local database, then

Calls the server_auths_client function to validate the login information

Calls the server_challenge function to begin bio authentication protocol

![Diagram](Matlab GUI Related Functions)

**Figure 40: Clientlogin structure**

Clientlogin contains the main MatLab functions for the graphical user interface. It has the definition of create and callback functions of the objects in the graphical interface, such as the push buttons, edit text boxes, and static text boxes. The create function of the objects sets its properties at the time the object is instantiated. The callback function is executed when an event happened to the object, such as a mouse click. In particular, the login push button callback function as shown on Figure-40 calls the major code blocks to simulate the dynamic authentication protocol.
As shown on Figure-41, the clientlogin flow started by converting to proper format the user input from the graphical user interface. Then, it updates the base MatLab workspace and calls clientdb function to load the client database. Once loaded, it then searches the login information in the database. If there’s no match, then it displays a login error message. If there is a match, then it calls the server_auths_client function to validate the login information against the server database. If there’s a match in the server side validation, then it calls the...
server_challenge function to begin the next part of the biometric dynamic authentication simulation. After executing the server_challenge function, it updates the graphical user interface.

**Block-2: server_auths_client**

The function of this code block is summarized as follows:

- Loads the server data by calling serverdb function
- Validates the login ID and password based on the contents of the server database
- If login information is not valid, it sends a message to the client and terminates the authentication protocol.

As shown on Figure-42, the server_auths_client function starts by getting the login information from the MatLab base workspace. Then, it loads the server database by calling serverdb function. Once loaded, it then searches the login information in the database. If there’s no match, then it displays a login error message. Finally, the MatLab base workspace is updated whatever the result of the search is.
Block-3: server_challenge

The function of this code block is shown in Figure-43 and summarized as follows:

Randomize the selection of challenge whether Iris or fingerprint

Generate a random index that will be asked from the client

Generate the server challenge packet

Calculate the MD5 hash of the server challenge packet

Encrypt the server challenge using a pre-shared key

Send the encrypted server challenge and the calculated MD5 hash

Call the client_response function

Figure 43: Server_challenge flow
Server challenge function first initializes its local variables by loading the values from the MatLab base workspace. It randomly selects the type of bio-template using a random number generator. The type can be an Iris or a fingerprint bio-template. Then, it generates the random challenge. Inside the random challenge is the asked index. The asked index is the index of the bio-template that the client should send back in its response. After the server challenge packet has been formed, its MD5 has is calculated and then encrypted using AES algorithm. These values are then uploaded to the MatLab base workspace.

**Block-4: client_response**

The function of this code block is shown in Figure-44 and summarized as follows:

- Receives the encrypted server challenge and its MD5 hash
- Decrypts the server challenge
- Calculates the MD5 hash of the decrypted server challenge
- Compares the received server challenge hash with the calculated hash of decrypted server challenge
- If hashes don’t match, it terminates the bio authentication protocol
- Generates the client response bio-template
- XOR the response with the received challenge (response XOR challenge)
- Calculates the MD5 hash of the (response XOR challenge)
- Encrypts the (response XOR challenge)
- Calls the server_response function
Client_response function first initializes its local variables by loading the values from the MatLab base workspace. Then it decrypts the server challenge. After decryption, it then calculates the MD5 hash. The calculated hash is compared to the hash in the server challenge. If the hashes do not match, then either the encrypted
message or the sent hash has been modified. In this case, it will display a login error message, send error message to the server, then update the MatLab base workspace for the result. Otherwise (hashes matched), it will continue and load the parsed client database in the MatLab base workspace. Then the asked bio-template is extracted to form the client response packet. An XOR operation is applied between server challenge and client response. The MD5 hash of the XOR operation packet is calculated. XOR value packet is encrypted and then the MatLab base workspace is updated.

**Block-5: server_response**

The function of this code block is shown in Figure-45 and summarized as follows:

- Receives the encrypted client (response XOR challenge) and its MD5 hash
- Decrypts the client (response XOR challenge)
- Calculate the MD5 hash of the decrypted client (response XOR challenge)
- Compare the calculated hash of the decrypted (response XOR challenge) and the received hash
- Terminate the authentication protocol if the hashes don’t match.
- Fetch the bio-template of the asked template from server database
- Calculate the Hamming distance of the received client response bio-template and the bio-template from the server database
- Decides whether to accept or reject the client based on the matching score threshold
Server_response function first initializes its local variables by loading the values from the MatLab base workspace. Then it decrypts the client response. After decryption, it then calculates the MD5 hash. The calculated hash is compared to the hash in the client response. If the hashes do not match, then either the encrypted
message or the sent hash has been modified. In this case, it will display a login error message, send error message to the client, then update the MatLab base workspace for the result. Otherwise (hashes matched), it will continue and load the parsed server database in the MatLab base workspace. Then the asked bio-template is extracted and the matching score is calculated between the received bio-template in the client response and the entry in the server database. If the matching score is less than the threshold, it will display a login error message, send error message to the client, then update the MatLab base workspace for the result. Otherwise, it will proceed and generate a successful login server response to the client, then update the MatLab base workspace for the result.

**Block-6: encrypt, decrypt**

The function of this code block is shown in Figure-46 and summarized as follows:

- Initializes the AES code
- Converts the input to proper format for the AES code
- Calls the AES code
- Clear all the AES variables

![Diagram](image)

Figure 46: Encrypt and decrypt flow
Encrypt and decrypt functions are the wrapper function of the AES block. These functions start by initializing the S-boxes and polynomial matrices. The input data for AES is formatted and converted from hexadecimal to decimal. After, it is fed to the AES block. The output of the AES block is reformatted back from decimal to hexadecimal and then returns the value to the calling function. All variables used in the calculation are then cleared.

**Block-7: serverdb, clientdb**

The function of this code block is shown in Figure-47 and summarized as follows:

- SERVERDB loads serverdb.csv
- CLIENTDB loads clientdb.csv
- Parses the contents of the csv file
- Puts the variables on the global Matlab workspace
- Clears all used variables
Figure 47: ServerDB and ClientDB flow

ServerDB and ClientDB are utility code blocks used to load the comma delimited CSV files. These files are the databases used in the simulation. ServerDB and ClientDB open its corresponding file and read it line by line until it encounters a non-string or non-character line. It interprets this as the end of file or end of the database. In each line, values are separated by a comma. The line string is split into an array with size based on the number of comma delimiters. The local variables are updated in every line parsed. When it encounters the end of the database, the loop breaks and then the MatLab base workspace is updated with the parsed values. ServerDB and ClientDB are almost identical except for the CSV files they are parsing. ServerDB parses serverdb.csv while ClientDB parses clientdb.csv.

Block-8: AES

Advance Encryption Algorithm 128 is based substitution –permutation network design principle as shown in Figure-48. It operates on a 4x4 column-major order matrix of bytes referred to as state. The round keys are derived from the cipher key using Rijndael’s key schedule. In the initial round, each byte of the state is combined with a block of the round key using bitwise xor. In the rounds prior to the final round, the state undergoes a non-linear substitution step where each byte is replaced with another according to a lookup Table (SubBytes). Then a transposition step where the last three rows of the state are shifted cyclically a certain number of steps (ShiftRows). And then it goes through a mixing operation which operates on the columns of the state, combining the four bytes in each column (MixColumns). The final round is similar to the previous round except there is no MixColumns. (Stalling, 2003) (Kumar and Tiwari, 2012).
Block-9: MD5

This code block calculates the MD5 hash as shown in Figure-49. After initialization, it makes sure that all inputs are ASCII characters. Then, the message is padded to make its length divisible by 512. The padded message is converted into 32-bit integer in little endian format. The 512-bit block is then processed using the MD5 algorithm (Stalling, 2003). The result is then converted back to hexadecimal format. The result is then returned to the calling function.
Figure 49: MD5 flow
6.4 Simulation Results

A result in one of the simulations is shown in Figure-50. The client fills up the login information edit boxes and clicks LOGIN push button. At the client, the login
information is verified onto the local client database and check if the user id exists and the corresponding password is correct. The login information is also verified on the server side. If the login is successful, a message is sent to the client. The server then generates a challenge to the client. The client responds by sending the asked bio-template to the server. The server verifies the client response by comparing it with the server database using Hamming distance. The comparison generates a matching score. If the matching score is more than 95%, then the client is successfully authenticated as shown in the Figure above. In addition, the client can regenerate its response (this is similar to scanning your finger print or Iris again) and resend it by clicking Rand.Auth. Push button. However, the client can only resend its response only three times. Otherwise, the authentication completely fails. A timer is also added to measure the duration of each simulation.

![Runtime vs. Database Size](image)

Figure 51: Runtime vs. Database size (in bytes)

The comparison of runtime versus increasing size of database is done as shown in Figure-51. As the number of entries in the database increases, the runtime also increases. This corresponds to the time it takes to search for the user and extract the
associated bio-template information. Although the type of database might affect the
performance, the runtime trend will stay in a similar increasing pattern since all the
other calculation in the authentication protocol, such as the AES and MD5
calculation remains the same.
Chapter 7: Security Validation of Proposed Protocol

7.1 Scyther Tool Briefing

Designer of any protocol should ensure that the protocol achieves the requirements of performance, scalability and security. The performance and scalability analysis was done using Matlab simulation. In this part; the security of the proposed protocol is validated using Scyther tool. Scyther is a tool used for the formal analysis of security protocols against the known attacks, such as replay, masquerade, modification and unauthorized access. The tool is assuming that, any cryptographic function is perfect and the adversary cannot decrypt the massage unless he knows the secret key. Scyther tool is used to find problems that are related to the nature of the protocol design which can help the designer to validate his protocol. The tool is compatible with both MAC and windows operating systems. Scyther tool is coded using Secure Protocol Description Language (SPDL). The purpose of this language is to describe the security properties of the protocol (Cremers and Mauw, 2012).

Validating the security of the transferred challenge and response is the focal point in identifying the security of the proposed protocol. The proposed protocol has two agents (parties) which are: client and server. The protocol is consists of set of roles as shown in Figure-52. The roles also is consists of sequence of events. The events are either sent, received or match items between the two parties as shown in Figure-53. The scythe language also consists of certain atomic items, such as freshly generated values, variables, symmetric key, asymmetric keys and hash functions (Scyther manual).
Scyther tool is designed to measure the Secrecy and Authentication of the evaluated protocol. The evaluation functions are done by Claims functions. Claims functions are used to model any intended security property as shown in Figure-54.
According to (Cremers and Mauw, 2012) Secrecy expresses that certain information is not revealed to an adversary, even though this data is communicated over an un-trusted network. In Scyther tool the evaluation of secrecy depends on two security properties:

**Secret:** It ensures that each event achieve Secrecy.

**SKR:** It ensures that each event achieve Secrecy but using a Secret Key.

(Lowe, 1997) said that: “the appropriate authentication requirement will depend upon the use to which the protocol is put, and identify several possible definitions of “authentication”. The authentications definitions are considering a protocol that aims to authenticate a responder B (Receiver) to an Initiator A (Sender). The following forms are defined:

(Aliveness): “The protocol guarantees to an initiator A aliveness of another agent B if, whenever A (acting as initiator) completes a run of the protocol, apparently with responder B, then B has previously been running the protocol “ (Lowe, 1997).
(Weak agreement): “The protocol guarantees to an initiator A weak agreement with another agent B if, whenever A (acting as initiator) completes a run of the protocol, apparently with responder B, then B has previously been running the protocol, apparently with A (Lowe, 1997).

(Non-injective agreement). “The protocol guarantees to an initiator A non-injective agreement with a responder B on a set of data items ds (where ds is a set of free variables appearing in the protocol description) if, whenever A (acting as initiator) completes a run of the protocol, apparently with responder B, then B has previously been running the protocol, apparently with A, and B was acting as responder in his run, and the two agents agreed on the data values corresponding to all the variables in ds” (Lowe, 1997).

(Non-injective synchronization) This property requires that there exists a cast function such that for all claim run events whose intended partners are honest, certain events occur in the trace (Cremers and Mauw, 2012).

(Agreement over data) It shows that the role I agrees with the role R on a set of terms. It is done in Scyther by using the word Commit (Cremers and Mauw, 2012).

7.2 The Security Validation
The proposed authentication protocol is validated using scythe tool. Two roles are created: Client and Server. The challenge and response packets are evaluated in order to ensure all levels of secrecy and authentication are met. Figure-55 presents the written scythe code and Figure-56 presents the validation of all security properties which proofs that the all security requirements are met.
Figure 55: Scyther written code of the proposed protocol
Figure 56: Security properties validation of the proposed protocol
Chapter 8: Conclusion

This thesis discusses a novel secure protocol for continuous and dynamic authentication of session’s users using biometrics data. It proposes a biometrics-based secure authentication protocol against physical session hijacking attacks. The authentication process is based on user’s biometrics, mainly fingerprint and Iris. The standard digital representations in the databases of single fingerprint data and Iris image are discussed in detail. The required biometrics portion for authentication is embedded in the IP header’s Option field to ensure compatibility with different TCP/IP protocols implementation. The security of the proposed protocol is validated using Scyther tool in terms of authenticity and secrecy. The proposed protocol is simulated using Matlab tool. The simulation shows that the proposed protocol requires larger run time than the ordinary login/password scheme, but it is acceptable due to the security requirements achieved.

In future work, this protocol can be modified to include other types of biometrics. The attacks against the proposed attacks will be investigated. Also, common service authentication based on the proposed protocol will be implemented and evaluated, such as HTTP, FTP or Telnet.
Bibliography


Iso/iec 19794-2:2005 finger minutiae data.

Iso/iec 19794-6:2005 Iris image data.


### Appendix

The header of single finger record

**Table 8: Detailed structure of single finger record header**

<table>
<thead>
<tr>
<th>Field</th>
<th>Size</th>
<th>Valid Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finger Position</td>
<td>1 byte</td>
<td>0 to 16</td>
<td>See Table 3</td>
</tr>
<tr>
<td>View Number</td>
<td>4 bits</td>
<td>0 to 16</td>
<td></td>
</tr>
<tr>
<td>Impression Type</td>
<td>4 bits</td>
<td>0 to 3 or 0</td>
<td>See Table 4</td>
</tr>
<tr>
<td>Size of Scanned Image in X-Dir</td>
<td>2 bytes</td>
<td>0x0000 - 0xFFFF</td>
<td>in pixels</td>
</tr>
<tr>
<td>Size of Scanned Image in Y-Dir</td>
<td>2 bytes</td>
<td>0x0000 - 0xFFFF</td>
<td>in pixels</td>
</tr>
<tr>
<td>X (horizontal) Resolution</td>
<td>2 bytes</td>
<td>0x0000 - 0x7FFF</td>
<td>in pixels per cm</td>
</tr>
<tr>
<td>Y (vertical) Resolution</td>
<td>2 bytes</td>
<td>0x0000 - 0x7FFF</td>
<td>in pixels per cm</td>
</tr>
<tr>
<td>Finger Image Quality Number of quality blocks</td>
<td>1 byte</td>
<td>0-10</td>
<td>A value of 0 indicates that there are no 8-byte quality blocks present; 255 indicates quality not reported; 254 indicates quality failed to compute</td>
</tr>
<tr>
<td>Quality value</td>
<td>1 byte</td>
<td>0 - 100, 254, 255</td>
<td></td>
</tr>
<tr>
<td>Quality vendor ID</td>
<td>2 bytes</td>
<td>0x0000 - 0xffff</td>
<td>Vendor assigned</td>
</tr>
<tr>
<td>Quality algorithm ID</td>
<td>2 bytes</td>
<td>0x0000 - 0xffff</td>
<td>Vendor assigned</td>
</tr>
<tr>
<td>Minuta Field Len</td>
<td>4 bits</td>
<td>0 - 16</td>
<td></td>
</tr>
<tr>
<td>Ridge Ending Type</td>
<td>4 bits</td>
<td>0 - 16</td>
<td></td>
</tr>
<tr>
<td>Minuta Extraction Vendor ID</td>
<td>2 bytes</td>
<td>0x0000 - 0xffff</td>
<td>II registered with ISIA</td>
</tr>
<tr>
<td>Minuta Extraction Algorithm ID</td>
<td>2 bytes</td>
<td>0x0000 - 0xffff</td>
<td>II registered with ISIA</td>
</tr>
<tr>
<td>Number of Minutiae</td>
<td>1 byte</td>
<td>0 - 65,535</td>
<td>Each pair of characters coded in 8 bits</td>
</tr>
<tr>
<td>Capture Date Time</td>
<td>10 bytes</td>
<td>YYYYMMDDhhmmss</td>
<td></td>
</tr>
<tr>
<td>Capture Device Vendor ID</td>
<td>2 bytes</td>
<td>0x0000 - 0xffff</td>
<td>Vendor specified</td>
</tr>
<tr>
<td>Capture Device Type ID</td>
<td>2 bytes</td>
<td>0x0000 - 0xffff</td>
<td>Vendor specified</td>
</tr>
<tr>
<td>Capture Equipment Certification</td>
<td>1 byte</td>
<td>0x0000 - 0x00FF</td>
<td>Certification organization</td>
</tr>
<tr>
<td>X (minutiae type in upper 2 bits)</td>
<td>2 byte</td>
<td>0 - 255</td>
<td>Each pair of characters coded in 8 bits</td>
</tr>
<tr>
<td>Y (upper 2 bits reserved)</td>
<td>2 byte</td>
<td>0 - 255</td>
<td>Each pair of characters coded in 8 bits</td>
</tr>
<tr>
<td>Angle θ</td>
<td>1 byte</td>
<td>0 to 255</td>
<td>Resolution is 1.40625 degrees</td>
</tr>
<tr>
<td>Minuta Quality</td>
<td>0 or 1 byte</td>
<td>0 to 100</td>
<td>For the 5-byte minutiae format there is no quality byte. For the 6-byte format the quality may range from 0 to 100 (255 indicates &quot;quality not reported&quot; and 254 indicates &quot;quality failed to compute score&quot;)</td>
</tr>
<tr>
<td>Extended Data Block Length</td>
<td>2 bytes</td>
<td>0x0000 - 0x00FF</td>
<td>0x0000 = no private area</td>
</tr>
<tr>
<td>Extended Data Area Type Code</td>
<td>2 bytes</td>
<td>0x0000 - 0x00FF</td>
<td>only present if Extended Data Element Length</td>
</tr>
<tr>
<td>Extended Data Area Length</td>
<td>2 bytes</td>
<td>0x0000 - 0x00FF</td>
<td>only present if Extended Data Element Length</td>
</tr>
<tr>
<td>Extended Data</td>
<td>12 bytes</td>
<td>0x0000 - 0x00FF</td>
<td>only present if Extended Data Element Length</td>
</tr>
</tbody>
</table>

Each extended data area may contain vendor-specific data, or one or more of the following (in any order):

- Ridge count extraction method: 1 byte, 0 to 2
- Ridge count data - dx #1: 1 byte, 1 to # of minutiae
- Ridge count data - dx #2: 1 byte, 1 to # of minutiae
- Ridge count data - dy: 1 byte, 1 to # of minutiae
- Additional ridge counts...

| Reserved | 4 bits | 0x00 |
| Reserved | 4 bits | 0x00 |
| X-location | 14 bits | 0x00 |
| Y-location | 14 bits | 0x00 |
| Reserved | 4 bits | 0x00 |
| Number of data pairs | 4 bits | 0 to 12 |
| X-location | 14 bits | 0x00 |
| Y-location | 14 bits | 0x00 |

Cell Width | 1 byte | 1 to 255
Cell Height | 1 byte | 1 to 255
Cell Information Bit Depth | 1 byte | 1 to 255
Cell Quality Data | CellData_0 |
### Iris Representation Header

Table 9: Detailed structure of Iris representation header

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Length</th>
<th>Valid values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Representation Length</td>
<td>4 bytes</td>
<td>63 to ((2^N) – 16)</td>
<td>The representation length field denotes the length in bytes of the representation including the representation header field.</td>
</tr>
<tr>
<td>2</td>
<td>Capture date and time</td>
<td>9 bytes</td>
<td>See ISO/IEC 19794-1, clause 12.3.2</td>
<td>The capture date and time field shall indicate when the capture of this representation started in Coordinated Universal Time (UTC). The capture date and time field shall consist of 9 bytes. Its value shall be encoded in the form given in ISO/IEC 19794-1.</td>
</tr>
<tr>
<td>3</td>
<td>Capture device technology identifier</td>
<td>1 byte</td>
<td>0 (0x00&lt;sub&gt;hex&lt;/sub&gt;): Unknown or Unspecified</td>
<td>The capture device technology ID shall be encoded in one byte. This field shall indicate the class of capture device technology used to acquire the captured biometric sample. A value of 0x00 indicates unknown or unspecified technology.</td>
</tr>
<tr>
<td>4</td>
<td>Capture device vendor ID</td>
<td>2 bytes</td>
<td>0000&lt;sub&gt;hex&lt;/sub&gt; (Unspecified) or Registered Value (IBIA or otherwise)</td>
<td>The capture device vendor identifier shall identify the biometric organisation that owns the product that created the BDR. The capture device vendor identifier shall be encoded in 2 bytes (0000&lt;sub&gt;hex&lt;/sub&gt; to FFFF&lt;sub&gt;hex&lt;/sub&gt;). A value of 0000&lt;sub&gt;hex&lt;/sub&gt; indicates that the capture device vendor is unspecified.</td>
</tr>
<tr>
<td>5</td>
<td>Capture device type ID</td>
<td>2 bytes</td>
<td>0000&lt;sub&gt;hex&lt;/sub&gt; (Unspecified) or Registered Value (IBIA or otherwise)</td>
<td>The capture device type ID shall identify the product type that created the BDR. It shall be assigned by the registered product owner or other approved registration authority. A value of all zeros shall indicate that the capture device type is unspecified.</td>
</tr>
<tr>
<td>6</td>
<td>Quality block</td>
<td>1 to n bytes</td>
<td>See ISO/IEC 19794-1</td>
<td>A quality record shall consist of a length field followed by zero or more quality blocks. The length field shall consist of one byte. It shall represent the number of quality blocks as an unsigned integer. Each quality block shall consist of a quality score, a quality algorithm vendor identifier, and a quality algorithm identifier. A quality score should express the predicted comparison performance of a representation. A quality score shall be encoded in one byte as an unsigned integer. Allowed values are 0 to 100 with higher values indicating better quality. IMAGE_QUAL_FAILED = 255 (FF&lt;sub&gt;hex&lt;/sub&gt;) indicating an attempt to calculate a quality score failed. The quality algorithm vendor identifier shall identify the product of the quality algorithm. The quality algorithm vendor identifier shall be encoded in two bytes carrying a CEFFF biometric organization identifier (registered by IBIA or other approved registration authority). A value of all zeros shall indicate that the quality algorithm vendor is unspecified.</td>
</tr>
<tr>
<td>7</td>
<td>Representation sequence number</td>
<td>2 bytes</td>
<td>1 to ... number of representations = 65 536</td>
<td>Representation sequence number</td>
</tr>
<tr>
<td>8</td>
<td>Eye label</td>
<td>1 byte</td>
<td>SUBJECT_EYE_LABEL_LEFT = 0 (0x00&lt;sub&gt;hex&lt;/sub&gt;)</td>
<td>These refer to the subject's own eyes.</td>
</tr>
<tr>
<td>9</td>
<td>Image type</td>
<td>1 byte</td>
<td>IMAGE_TYPE_UNCROPPED = 1 (0x01&lt;sub&gt;hex&lt;/sub&gt;)</td>
<td>An uncropped rectilinear iris image.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IMAGE_TYPE_VGA = 2 (0x02&lt;sub&gt;hex&lt;/sub&gt;)</td>
<td>A rectilinear iris image in VGA (640x480) format.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IMAGE_TYPE_CROPPED = 3 (0x03&lt;sub&gt;hex&lt;/sub&gt;)</td>
<td>A cropped, centred, Iris image with (0.6R, 0.2R) margins.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IMAGE_TYPE_CROPPED_AND_MASKED = 7 (0x07&lt;sub&gt;hex&lt;/sub&gt;)</td>
<td>A cropped and region-of-interest masked, centred, image with (0.6R, 0.2R) margins.</td>
</tr>
<tr>
<td>10</td>
<td>Image format</td>
<td>1 byte</td>
<td>IMAGEFORMAT_MONO_RAW = 2 (0x02&lt;sub&gt;hex&lt;/sub&gt;)</td>
<td>Format of image data. For all image types, see Table 9 and the normative requirements of the clauses cited.</td>
</tr>
<tr>
<td>Field Description</td>
<td>Data Type</td>
<td>Value Range/Details</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------</td>
<td>---------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iris image properties bit field</td>
<td>1 byte</td>
<td>Bits 1-2, i.e. least significant bits: ORIENTATION_UNDEF = 0, ORIENTATION_BASE = 1, ORIENTATION_FLIPPED = 2, Bits 3-4: ORIENTATION_UNDEF = 0, ORIENTATION_BASE = 1, ORIENTATION_FLIPPED = 2, Bits 5-6: 0, 0, Bits 7-8: PREVIOUS_COMPRESSION_UNDEF = 0, PREVIOUS_COMPRESSION_LOSSLESS_OR_NONE = 1, PREVIOUS_COMPRESSION_LOSSY = 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image width</td>
<td>2 bytes</td>
<td>Width in pixels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image height</td>
<td>2 bytes</td>
<td>Height in pixels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bit depth</td>
<td>1 byte</td>
<td>At least 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Range</td>
<td>2 bytes</td>
<td>2 to (2^{2n} - 2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roll angle of eye</td>
<td>2 bytes</td>
<td>(65.535 \times \text{angle}/360) modulo 65.535, where angle is measured in radians</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roll angle uncertainty</td>
<td>2 bytes</td>
<td>(65.535 \times \text{uncertainty}/180) where uncertainty is measured in degrees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iris centre, smallest X</td>
<td>2 bytes</td>
<td>(65.535 \times \text{coordinate}/100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iris centre, largest X</td>
<td>2 bytes</td>
<td>(65.535 \times \text{coordinate}/100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iris centre, smallest Y</td>
<td>2 bytes</td>
<td>(65.535 \times \text{coordinate}/100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iris centre, largest Y</td>
<td>2 bytes</td>
<td>(65.535 \times \text{coordinate}/100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iris diameter, smallest</td>
<td>2 bytes</td>
<td>(65.535 \times \text{coordinate}/100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iris diameter, largest</td>
<td>2 bytes</td>
<td>(65.535 \times \text{coordinate}/100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image length</td>
<td>4 bytes</td>
<td>Size of the image data (Representation body), in bytes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>