

國立交通大學

資訊學院 資訊學程

碩士論文

適用於各種企業應用之
高擴充性混合式指紋辨識系統



A Hybrid Fingerprint Identification System with High Scalability
for Various Enterprise Applications

研究生：王偉榕

指導教授：王國禎 博士

中華民國一〇一年七月

適用於各種企業應用之
高擴充性混合式指紋辨識系統

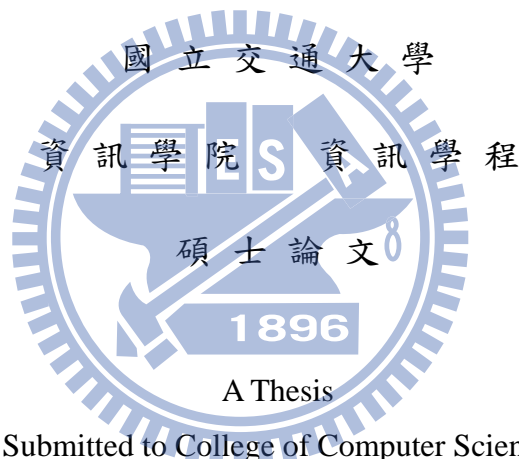
A Hybrid Fingerprint Identification System with
High Scalability for Various Enterprise Applications

研究生：王偉榕

Student : Wei-Jung Wang

指導教授：王國禎

Advisor : Dr. Kuochen Wang



Submitted to College of Computer Science
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Science
in
Computer Science
July 2012

Hsinchu, Taiwan, Republic of China

中華民國一〇一年七月

適用於各種企業應用之 高擴充性混合式指紋辨識系統

學生：王偉榕 指導教授：王國禎 博士

國立交通大學 資訊學院 資訊學程 碩士班



近年來，由於企業規模的迅速成長，在企業應用上能夠整合具有高安全性、低成本且可以快速認證使用者身份的方法顯得更加重要及需要。身份認證的目的在於個體的識別，除了傳統的密碼比對方法外，指紋辨識在市場上是一個普遍且可靠的生物辨識技術。為了符合企業應用的不同需求，我們提出一種混合式的指紋辨識系統，簡稱 HFIS。對於強調方便使用的企業應用，我們提出 HFIS-HTTP 來滿足使用者在任何平台、任何地方，只要透過網頁瀏覽器就可以經由認證使用公司內部資源；而對於強調能在短時間內處理大量需求的應用，我們所提出的 HFIS-TCP 可以滿足企業內部網路高安全性及快速反應的需求。在本論文中，我們以傳送指紋特徵

(feature-based), 而非指紋圖像 (image-based) 來降低指紋辨識所需的傳送資料量及傳遞時間。由於在指紋辨識系統內, 指紋特徵抽取是其中最耗時的工作項目, 我們將該項目程序由原本在指紋伺服器執行, 改為分散到各個客戶端的終端機執行, 以降低指紋伺服器的工作負載。在反應時間不超過兩秒的條件下, 與其他以傳送指紋圖像的方法比較, 在可支援的客戶端數量方面, HFIS-HTTP 在截止期限錯失率 2% 時, 比 Liu 的方法增加了 41.92%; 在截止期限錯失率 5% 時, 比 Wang 的方法增加了 84.06%。HFIS-TCP 在截止期限錯失率 3% 時, 比 Chang 的方法增加了 45.28%。至於平均反應時間則分別減少 23.66%、28.89% 及 14.58%。總而言之, 除了 HFIS-TCP 在較低的截止期錯失率 (0% 和 1%) 情況下, 我們提出的 HFIS-HTTP 和 HFIS-TCP 兩種方法, 在其他所有情況下都比相關方法有較短的平均反應時間。因此, 本論文所提出的混合式指紋辨識系統非常適用於需要高安全性、低成本、快速反應時間, 以及高擴充性且有不同需求的企業應用。

關鍵詞：企業應用、指紋辨識、高擴充性、反應時間、系統架構。

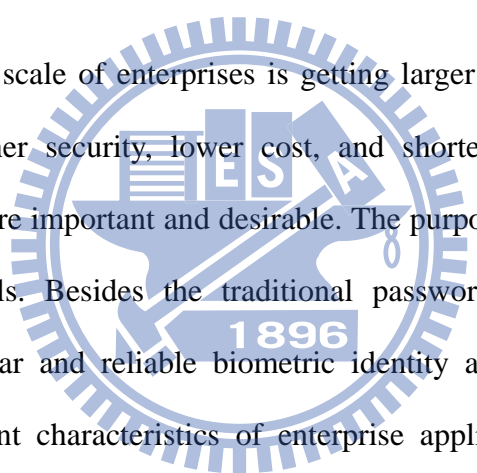
A Hybrid Fingerprint Identification System with High Scalability for Various Enterprise Applications

Student : Wei-Jung Wang Advisor : Dr. Kuochen Wang

Degree Program of Computer Science

National Chiao Tung University

Abstract



In recent years, the scale of enterprises is getting larger and the integration of identity authentication with higher security, lower cost, and shorter response time to enterprise applications becomes more important and desirable. The purpose of an identity authentication is to identify individuals. Besides the traditional password verification, the fingerprint identification is a popular and reliable biometric identity authentication technique in the market. To meet different characteristics of enterprise applications, we propose a hybrid fingerprint identification system (HFIS). For applications with emphasis on easy access, HFIS-HTTP is proposed to meet the requirement for users to access internal resources through web browsers from any platform, any place in Internet; for applications with emphasis on handling massive requests coming in a very short time, HFIS-TCP is proposed to meet high security and fast response time requirements in intranets. In our approach, we reduce the amount and time of data transmissions by transferring fingerprint features (feature-based) instead of fingerprint images (image-based). Since feature extraction is the most time-consuming task in fingerprint identification systems, we offload this task to each client to reduce the loading of the fingerprint server. Compared to other image-based approaches, under 2 seconds response time threshold, the proposed HFIS-HTTP increases the

number of clients supported by 41.92% (reduces average response time by 23.66%) compared to Liu's at 2% deadline miss ratio and by 84.06% (reduces average response time by 28.89%) compared to Wang's at 5% deadline miss ratio. In addition, the proposed HFIS-TCP increases the number of clients supported by 45.28% (reduces average response time by 14.58%) compared to Chang's s at 3% deadline miss ratio. In summary, the proposed HFIS-HTTP and HFIS-TCP have shorter average response time than related work for all cases, except HFIS-TCP with Embedded-based Clients at lower deadline miss ratios (0% and 1%). Therefore, the proposed HFIS is well suited for various enterprise applications which may require different combinations of high security, low cost, low response time, and high scalability.

Keywords: Enterprise application, fingerprint identification, high scalability, response time, system architecture.



Acknowledgements

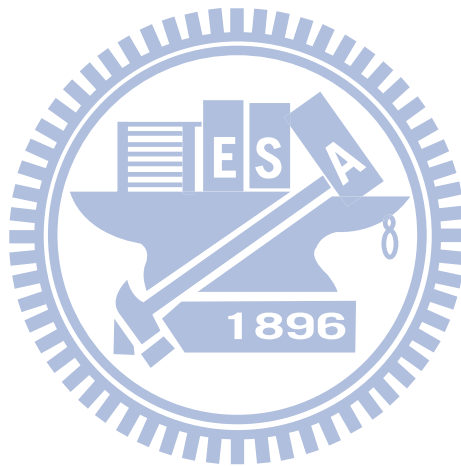
Many people have helped me a lot with this thesis. I deeply appreciate my thesis advisor, Dr. Kuochen Wang, for his intensive advice and guidance. I would also like to express my appreciation for the members of the *Mobile Computing and Broadband Networking Laboratory* (MBL) for their invaluable assistance and suggestions. Finally, I thank my family for their endless love and support.



Contents

Abstract (in Chinese)	i
Abstract	iii
Acknowledgements	v
Contents	vi
List of Figures	viii
List of Tables	ix
Abbreviations List	1
Chapter 1 Introduction	2
Chapter 2 Background and Related Work	5
2.1 Overview of fingerprint identification systems (FIS).....	5
2.2 RS485-based FIS versus Ethernet-based FIS	7
2.3 Related work of Ethernet-based FIS.....	11
Chapter 3 Proposed Hybrid Fingerprint Identification System	16
3.1 System architecture	16
3.1.1 Fingerprint Server.....	17
3.1.2 Database Server	18
3.1.3 TCP/IP Server.....	19
3.1.4 Web Server	19
3.1.5 Embedded-based Client.....	20
3.1.6 PC-based Client	21
3.2 Traffic reduction	22
3.3 Offloading.....	25

3.4 Features of our design approach.....	25
Chapter 4 Evaluation and Discussion.....	27
4.1 Simulation setup	27
4.2 Simulation results and discussion.....	28
Chapter 5 Conclusion.....	33
5.1 Concluding remarks.....	33
5.2 Future work	33
References.....	35

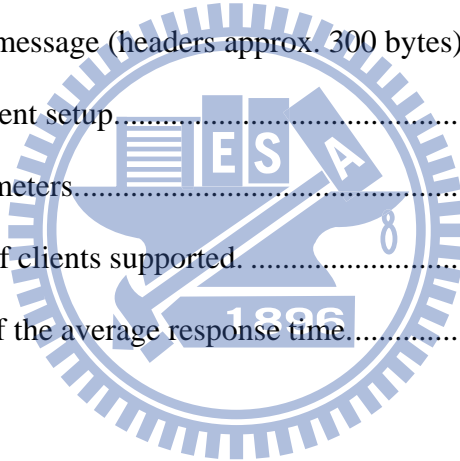


List of Figures

Figure 1. Fingerprint features classification.	5
Figure 2. Fingerprint feature characteristics.	5
Figure 3. Fingerprint identification process.	6
Figure 4. Feature extraction steps.	6
Figure 5. Data rate vs. cable length for RS-485 [18].	8
Figure 6. RS485-based FIS.	9
Figure 7. Embedded-based FIS [20] [22] [24].	12
Figure 8. PC-based FIS [19] [21] [23].	12
Figure 9. Proposed HFIS system architecture.	17
Figure 10. Fingerprint Server architecture.	18
Figure 11. Hardware architecture of LwFD.	20
Figure 12. Windows application design for PC-based Client.	21
Figure 13. Overhead comparison of HTTP-based and TCP/IP-based packet headers.	22
Figure 14. Deadline miss ratios of different FIS approaches.	29
Figure 15. Average response time under different deadline miss ratios.	31

List of Tables

Table 1. Biometric identification techniques [2].....	3
Table 2. Comparison of commonly used biometric identification methodologies in terms of High (H), Medium (M), and Low (L) [17].	3
Table 3. Comparison of Ethernet-based FIS and RS485-based FIS.	10
Table 4. Comparison of Embedded-based Client and PC-based Client.....	14
Table 5. Comparison of different FIS approaches.	14
Table 6. HTTP request message (headers approx. 600 bytes).	23
Table 7. HTTP response message (headers approx. 300 bytes).....	24
Table 8. Server environment setup.....	27
Table 9. Simulation parameters.....	28
Table 10. Extra number of clients supported.	30
Table 11. Improvement of the average response time.....	32



Abbreviations List

The list contains the main abbreviations used throughout this thesis.

FIS: Fingerprint Identification System

JDBC: Java Data Base Connectivity

API: Application Program Interface

SQL: Structured Query Language

IT: Information Technology

XML: Extensible Markup Language



Chapter 1

Introduction

Due to highly networking environments in enterprises, there are more and more enterprise applications developed for Internet use. Each application has its own mechanism of identity authentication to control who can use it, what can be used, and where one can use it. The traditional identity authentication is mainly to identify individuals by verifying users' passwords. However, it is obviously vulnerable and not safe. Therefore, a variety of biometric identification methods has been presented and commercialized recently. All these methods aim to offer alternative identity authentication methods for high security requirements.

Identification systems based on biometrics are capable of identifying individuals based on human characteristics, either physical or behavioral characteristics [1]. These characteristics are unique, measurable, identifiable, and verifiable. Users' feature information are created and stored in identification systems by sampling, extracting, and digitalizing [13]. By comparing the features information stored in a system with the one's present characteristics, the system can identify a person and decide to accept or reject such a person. Biometric identification has attracted increased attention as a means of reliable identity authentication. Today, there are many biometric identification techniques available to identify individuals [2], as shown in Table 1.

Table 1. Biometric identification techniques [2].

Physical characteristics	Behavioral characteristics
Fingerprint recognition	Keystrokes dynamics
Face recognition	Voice pattern recognition
Hand geometry recognition	Signature dynamics
Palm print recognition	
Iris recognition	
Retina recognition	
Vein pattern recognition	

A practical biometric identification methodology should satisfy the conditions of accurate identification and high performance with reasonable system requirements, such as safe to users, and easy to use by general people [17] [25], as shown in Table 2. Each biometric methodology has its own strengths and weaknesses and is suitable to different applications. There is no perfect method to meet the requirements of all applications but a balance and generally acceptable one widely used nowadays is the fingerprint identification [10] [14].

Table 2. Comparison of commonly used biometric identification methodologies in terms of High (H), Medium (M), and Low (L) [17].

Biometric identifier	Universality	Distinctiveness	Permanence	Collectability	Performance	Acceptability	Circumvention
Fingerprint	M	H	H	M	H	M	M
Face	H	L	M	H	L	H	H
Hand Geometry	M	M	M	H	M	M	M
Hand/Finger Vein	M	M	M	M	M	M	L
Iris	H	H	H	M	H	L	L
Signature	L	L	L	H	L	H	H
Voice	M	L	L	M	L	H	H

In this thesis, we propose a Hybrid Fingerprint Identification System (HFIS) that includes Fingerprint Server, Database Server, TCP/IP Server, Web Server, Embedded-based Client, and PC-based Client. To meet different characteristics of enterprise applications, two architectures, HFIS-HTTP and HFIS-TCP, are proposed, which are HTTP-based and TCP/IP-based respectively. HFIS-HTTP is for applications targeted for easy access. Users can access internal resources easily through web browsers from any platform, any place. HFIS-TCP is for applications that can handle massive requests in a very short time to meet high security and fast response requirements. Clients of the HFIS are in charge of image acquisition and feature extraction and then transfer the features to Fingerprint Server for matching. Feature extraction on each client also brings offloading issue. Evaluation results show that the proposed HFIS can increase the number of clients supported and reduce average response time.

This rest of this thesis is organized as follows. In Chapter 2, fingerprint identification systems are overviewed and existing approaches are reviewed. Chapter 3 presents the system architecture and the detailed design approach. Evaluation results between the proposed approaches and related work are discussed in Chapter 4. Finally, in Chapter 5, we give concluding remarks and future work.

Chapter 2

Background and Related Work

2.1 Overview of fingerprint identification systems (FIS)

Several years ago, fingerprint identification systems (FIS) were very expensive and only adopted in governments or some highly secure organizations. Due to the recent advanced semiconductor technology [7], various high density image sensors trigger more and more secure FIS's development. The image sensors can be thermal, pressure, optical or capacitive type and they are used to get a good quality of fingerprint image. Fingerprint features (minutia) can be found from fingerprint images and can be classified into several types, as shown in Figure 1. A fingerprint identification algorithm will record each feature by its type, orientation, spatial frequency, angle, position, and so on, as shown in Figure 2.

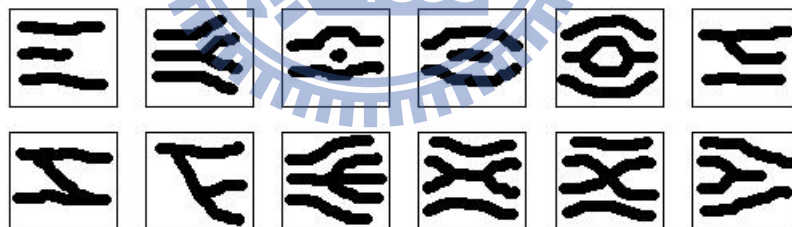


Figure 1. Fingerprint features classification.

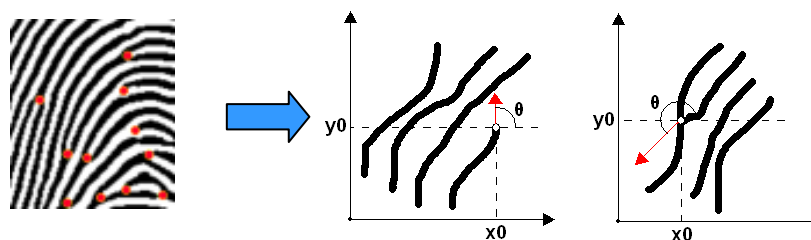


Figure 2. Fingerprint feature characteristics.

In Taiwan, we follow the standard from FBI in 1973 that describes an individual can be identified by at least 12 features. The fingerprint identification process is illustrated as Figure 3. The process is started at the image acquisition and ended at matching. The most time-consuming task of the fingerprint identification process is the feature extraction which consists of many image processing and information extraction as Figure 4.

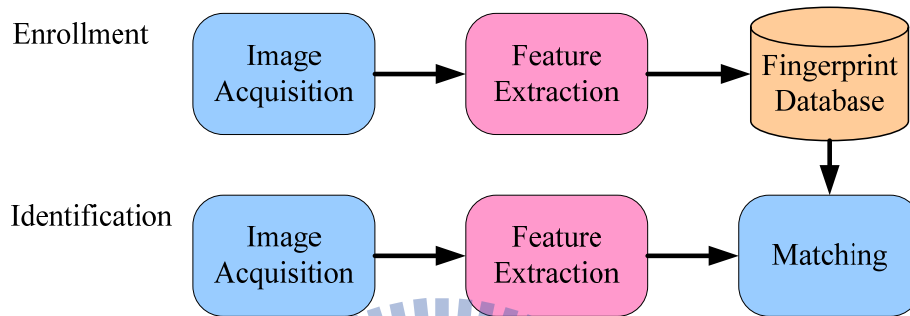


Figure 3. Fingerprint identification process.

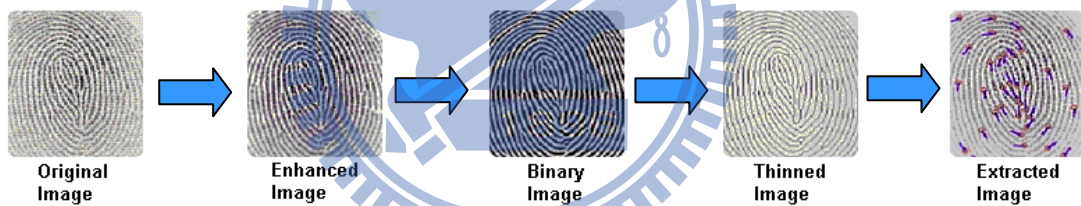


Figure 4. Feature extraction steps.

A traditional Fingerprint Identification System (FIS) is established by collecting fingerprint features of all users first, and then do real-time verification or identification by the extracted feature from the image captured by Embedded-based Client [20] [22] or PC-based Client [3] [9] [19] [21]. The architecture design of the traditional FIS is always for specific purpose applications only and the clients are in charge of image acquisition and transferring image to server for feature extraction and matching with HTTP or TCP/IP protocol. With the client requests increased, the server loading should be considered to keep balance for good performance. The requirements for a fingerprint identification system are fast response and

low cost without sacrificing its accuracy and security [8] [9]. Therefore, the trade-offs between the cost and performance should be taken into consideration and also depends on the purpose for commercial or industrial use. Recently, a multimodal identification system has been proposed with more than one type biometric technologies integrated to increase the security level [4] and some portable devices have been also integrated with fingerprint identification for access control authentication [5] [6] [8] [11] [12].

2.2 RS485-based FIS versus Ethernet-based FIS

From the view of the communication medium, there are two methodologies for FIS development and deployment, the RS485-based FIS and Ethernet-based FIS. Here, we explore why the RS485-based FIS is not suitable for the needs of enterprise applications and Ethernet-based FIS is the trend.

RS485 protocol is always implemented by polling handshaking. The data transmission rate ranges from 10 Kbps to 10 Mbps which is decided by the length of deployed cable from a device to host. The longer the cable is, the slower the data rate will be. Figure 5 shows the relation between the cable length and the data rate [18]. Due to the characteristics of the inverse proportion of the data rate and cable length, we can set 10 Kbps as the best transfer rate for maximum cable length. However, to guarantee the signal quality, 9600 bps is the most stable setting. Slow transmission rate results in the low real-time response. The error detection and correction in RS485 is also not easy to implement and the way to handle the error is usually to ask the device to resend the data, which is a simple and reliable, but time-consuming mechanism, especially for large data frame transmission. Therefore, the network scale is not easy to extend due to the signal characteristics. For RS485-based FIS, it always needs to find the balance among number of terminal node, line loading, cable length, and transmission rate.

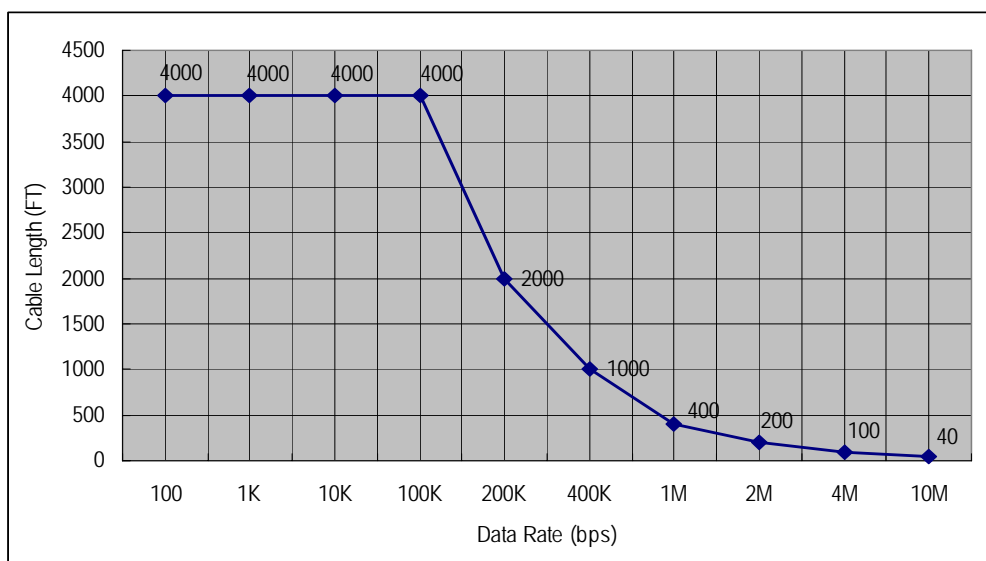


Figure 5. Data rate vs. cable length for RS-485 [18].

From Figure 6, we can see that the basic architecture of the RS485-based FIS. The RS485 host is an arbiter who is in charge of the polling process in the RS485 network. The host can be a PC which links to the RS485 network through the RS485 to RS232 converter. There are some solutions that combine the host with the converter as an embedded host controller, in which RS485 and Ethernet interfaces are integrated. In the deployment, the RS485 hub may need to ensure the signal quality in case the RS485 cable is too long to transmit good signals. In order to integrate RS485 network and Ethernet network, it needs at least three physical parts for data communication, access control device, RS485 host, and the central server. There exists small groups of RS485 network under each RS485 host and devices in different groups cannot talk to each other without the hosts' negotiation. A host has to manage the devices of its group by polling command with specific device ID. It means that the behavior of each device is passive and also brings the drawback of long response time.

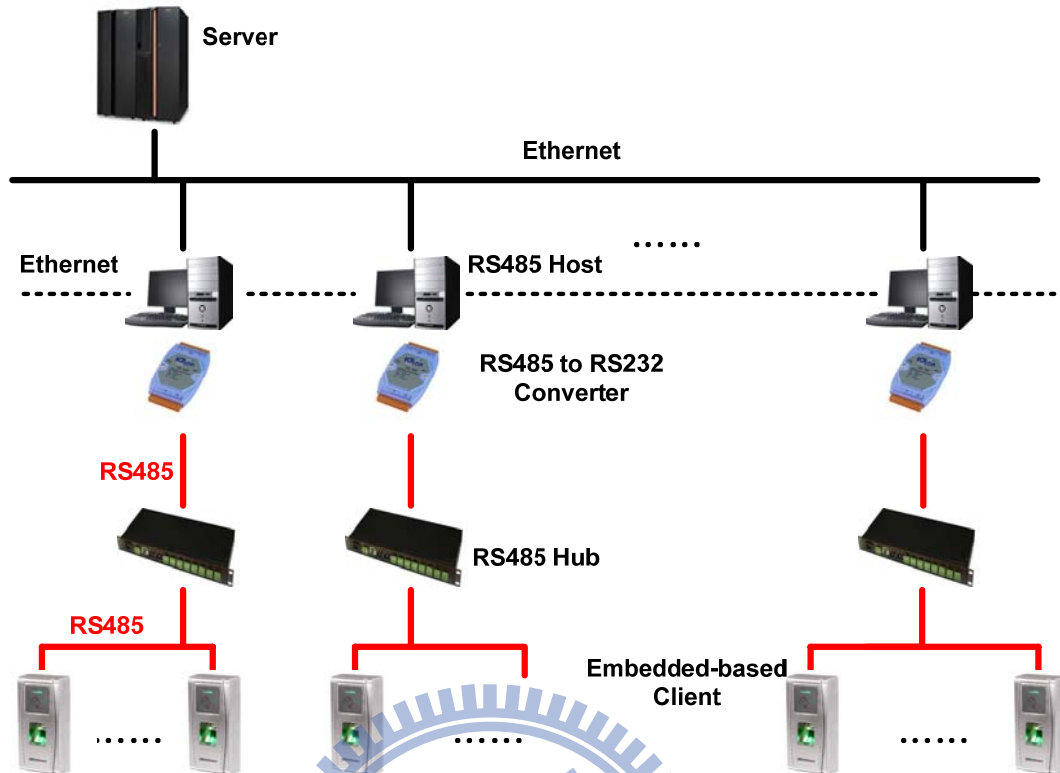


Figure 6. RS485-based FIS.

Once the RS485 network deployment is done, it will be very difficult to modify or extend its scalability. All the devices on the same cable generate a fixed loading which affects the signal level and the cable length. The environment interference is also an issue for system performance tuning. When one of the devices fails, it will result in all the group failure. This failure cannot be fixed by just removing the failure node, because the change of cable loading makes the signal level changed. Therefore, not only the deployment but also the maintenance is very tough to use RS485 network for large scale application.

We can know that the RS485 network is very suitable for small data frame transmission, like status monitoring, card reader application, and many simple control data transmission application. However, for large scale network and large data frame transmission requirements, it has many difficulties to guarantee its feasibility including the deployment issue, maintenance issue, and performance issue. Consequently, Ethernet-based FIS is the trend of fingerprint identification system that communicates directly on the Ethernet and it is very

feasible for applications development without so many environmental limitations. Based on different client type, Ethernet-based FIS has two classifications, Embedded-based FIS and PC-based FIS. Both are cooperating with a back-end server for fingerprint feature extraction and matching. Table 3 shows the comparison of the Ethernet-based FIS and RS485-based FIS by some critical terms.

Table 3. Comparison of Ethernet-based FIS and RS485-based FIS.

Aspect	Ethernet	RS485
Number of Node (Scalability)	Unlimited	< 64
Interoperability	Yes	No
Real Time	High, by active transmission	Low, by polling
Transmission Rate	10/100 Mbps	10 Kbps at 1200 m Normally 9600 bps, Max. 10 Mbps
Fault Tolerance	Fault detection and Correction	No
Transmission Failure Rate	Very low	High
Transmission Distance	Unlimited	< 1200 m (4000 ft)
Testing/ Failure Recovery/ Maintenance	Easy	Difficult

2.3 Related work of Ethernet-based FIS

The architecture of Ethernet-based FIS is client/server model which consists of a Fingerprint Server and one or many clients of Embedded-based or PC-based. A Fingerprint Server can be responsible for the fingerprint feature extraction, fingerprint matching and data management [22] [24] or only for data management [20]. The Embedded-based Client or PC-based Client is equipped with a fingerprint scanner being a simply fingerprint image acquisition device and transfers the image to Fingerprint Server to perform the identity authentication process [22] [24]. Embedded-based Client and PC-based Client can even be a role of matching device [20], but the performance highly depends on the client capability. The communication protocol between Fingerprint Server and Embedded-based Client/PC-based Client is based on TCP/IP or HTTP. TCP/IP protocol is usually used for Embedded-based FIS and HTTP protocol is used for browser/server mode as a 3-tier structure with Web Server being the middle bridge between Fingerprint Server and clients.

We reviewed Shi [20], Chang [22] and Li [24] as Embedded-based FIS. Shi [20] proposed the one-to-many (1-N) client matching with fingerprint feature database preloaded in each client. The extraction and matching is performed at the client that means there is no data transmission of request and response needed for identity authentication. The matching speed can be very fast if the computing power of the Embedded-based Client is high enough or the number of features to be matched does not exceed the client's capability. Chang [22] and Li [24] both adopt the one-to-many (1-N) server matching. The server involves the feature extraction of the real-time fingerprint image transferred from client and the comparison of the extracted feature with all features in the database. System performance depends on server capability and the efficiency of the image data transmission. Figure 7 shows the system architecture of the Embedded-based FIS.

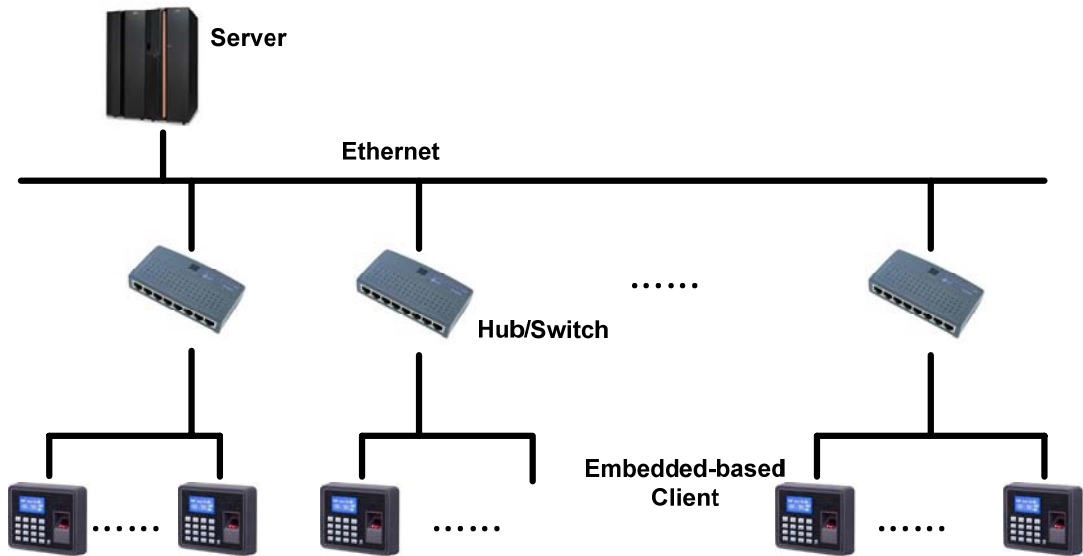


Figure 7. Embedded-based FIS [20] [22] [24].

The architecture of PC-based FIS is shown as Figure 8. The main different between Embedded-based FIS and PC-based FIS is the client type. With such a deployment, the mentioned related work can be classified into three matching methodologies, one-to-many (1-N) server matching [19], one-to-one (1-1) server matching [21] and one-to-one (1-1) client matching [23].

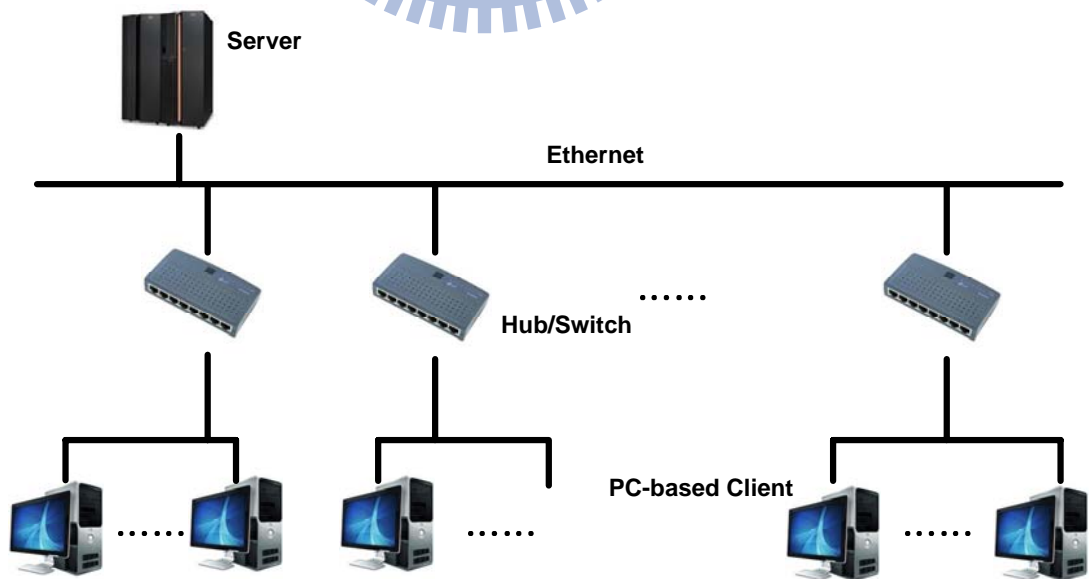


Figure 8. PC-based FIS [19] [21] [23].

In Liu [19], the fingerprint image is obtained from PC-based Client and transferred to Fingerprint Server by HTTP protocol for feature extraction and one-to-many matching. The fingerprint image is transferred with the corresponding user ID to Fingerprint Server for one-to-one matching in Wang [21]. With this additional user ID as the database index, the matching speed can be very fast, but the extra user-client interaction is needed. For these two server matching methods, the critical overhead to affect the system performance is the fingerprint image size, the HTTP headers of application layer packet and the extra TCP headers from data segmentation.

As to the one-to-one client matching of Shafi [23], the PC-based Client transfers the specific user ID as index to server and requests for the corresponding fingerprint feature stored in the database. So that the client can perform the matching with the feature extracted from user's present fingerprint image and the feature requested from the server. This kind of client matching is very efficient with feature transmission especially by TCP/IP protocol. The extraction and matching on each client can also lower the computational load of Fingerprint Server. However, the cost of the PC-based Client is very high when the high scalability is required.

We also compare PC-based Client and Embedded-based Client in some important terms. A special purpose Embedded-based Client is much cheaper than a PC-based Client. The size of the Embedded-based Client can be as compact as possible and the power consumption is also lower. The computational power of a PC-based Client is no doubt much higher than the Embedded-based one. Therefore, the time spent for feature extraction of a PC-based Client is shorter and the efficiency of one-to-many matching on client can achieve users' expectations. Table 4 shows the comparison of Embedded-based Client and PC-based Client.

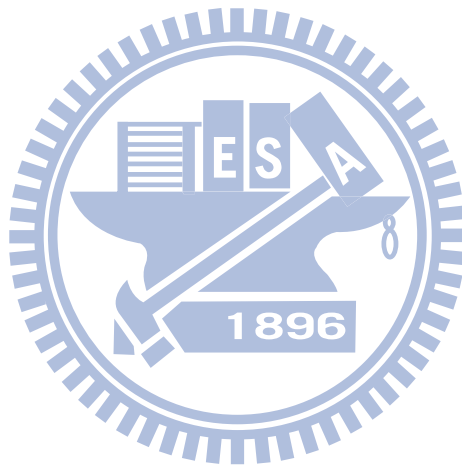
Table 4. Comparison of Embedded-based Client and PC-based Client.

Aspect	Embedded-based Client	PC-based Client
Cost	Low	Higher
Size	Compact	Big
Power Consumption	Low	High
Computational Power	Low	High
Feature Extraction Speed	Normal	Fast
Client Matching	Verification (1-1)	Identification (1-N)/ Verification (1-1)

Table 5. Comparison of different FIS approaches.

Approach	<i>HFIS (proposed)</i>	<i>Shi[20]</i>	<i>Chang[22]</i>	<i>Wang[21]</i>	<i>Liu[19]</i>
Network Transport	Hybrid (HTTP / TCP)	TCP	TCP	HTTP	HTTP
Client Type	PC/ Embedded	Embedded	Embedded	PC	PC
Feature Extraction	Client	Client	Server	Server	Server
Matching	Server 1-N	Client 1-N	Server 1-N	Server 1-1	Server 1-N
Fingerprint Data Transfer	Feature	None	Image	Image	Image
Overhead	Fingerprint Server Complexity	Data Sync	TCP header Image size	HTTP & TCP headers, Image size	HTTP & TCP headers, Image size
Response Time	Lower	Long	Medium	Medium	Low
Scalability	High	Low	Low	High	High
Cost	Low	Low	Low	High	High

Table 5 lists the comparison of the mentioned FIS approaches. In terms of network transport, client type, feature extraction, matching, fingerprint data transfer, overhead, response time, scalability and cost, we make comparisons among the Liu [19] and Wang [21] that we have mentioned as PC-based FIS, and the Embedded-based FIS: Shi [20], Chang [22] and the proposed HFIS.



Chapter 3

Proposed Hybrid Fingerprint Identification System

3.1 System architecture

We propose a hybrid fingerprint identification system, named HFIS, for HTTP-based (HFIS-HTTP) and TCP-based (HFIS-TCP) architectures. The proposed HFIS consists of the PC-based Client or Embedded-based Client as access control devices and the backend servers include the Fingerprint Server, Database Server, TCP/IP Server and Web Server. HFIS-HTTP is an FIS with features transferred by using the HTTP protocol for the browser/server model. It is suited for Internet-based environments. HFIS-TCP is an FIS with features transferred by using the TCP protocol for intranet applications. TCP-based proprietary protocols have been designed for higher security between Embedded-based Client and TCP/IP Server. They are very suited for Intranet-based environments. Figure 9 illustrates the system architecture of our proposed HFIS.

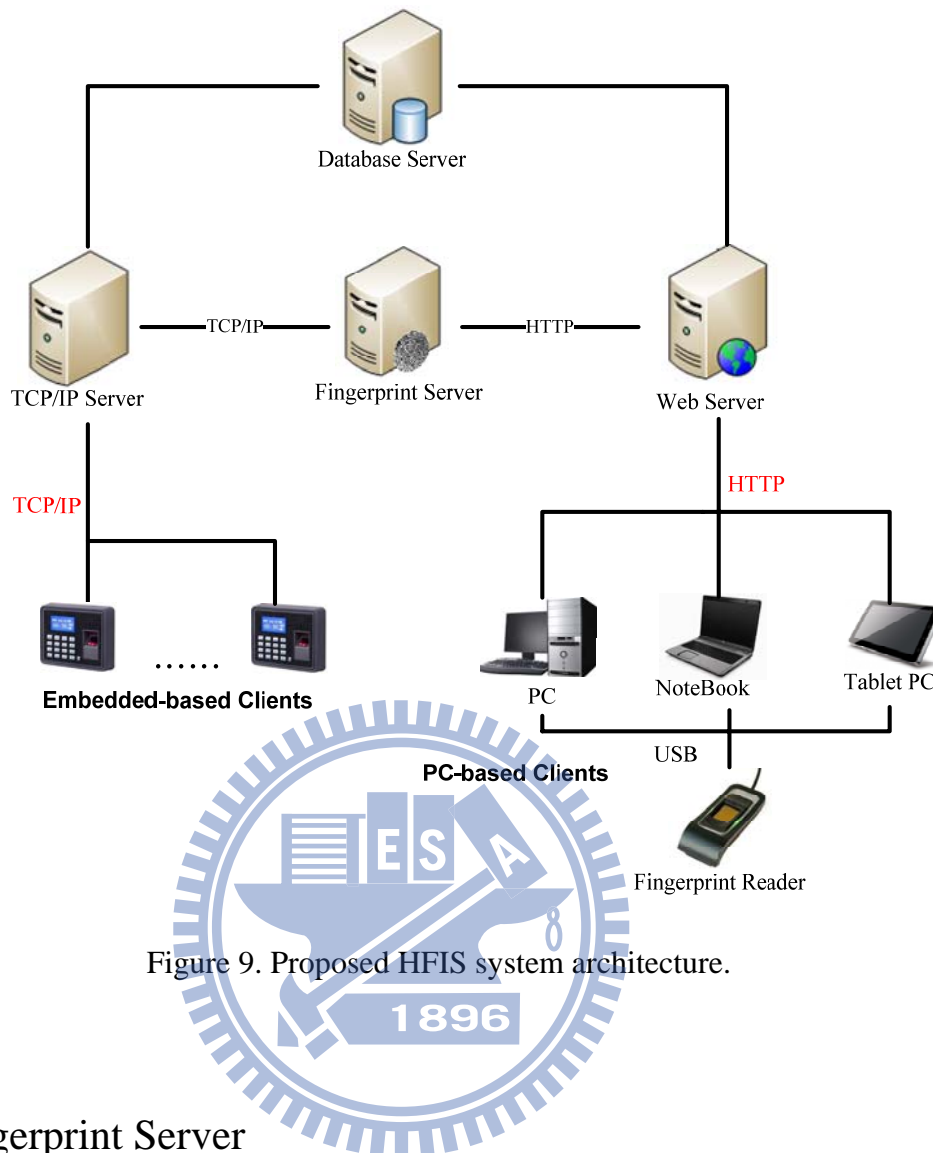


Figure 9. Proposed HFIS system architecture.

3.1.1 Fingerprint Server

Fingerprint Server is the critical part of the proposed HFIS. Its system architecture is shown in Figure 10. The application of fingerprint identification is developed by C language and running on an Ubuntu Linux with Intel Xeon 2.0GHz CPU×4 and 4GB RAM. It is mainly in charge of the requests of fingerprint extraction and matching from TCP/IP Server and Web Server which means that two specific socket listeners on two different ports are necessary, say, port 80 for HTTP packets and port 1500 for TCP packets.

As to the capability of the server design, it handles all the incoming requests by multithread processes. The maximum concurrent threads allowed in a Linux operating system is 1024, but some are used for system itself. Therefore, when all the processes are occupied,

Fingerprint Server will reply an error message “Exceed the max concurrent” to the middleware in Web Server or TCP/IP Server. Fingerprint Server also involves in the cache design. When Fingerprint Server boots, all fingerprint features stored in the database will be written into the cache with user ID as index for matching, so that it can greatly reduce the data fetch time from database. By such a design, the matching speed can reach around 200,000 fingerprint features per second, and the maximum capacity of features in the cache is 500,000.

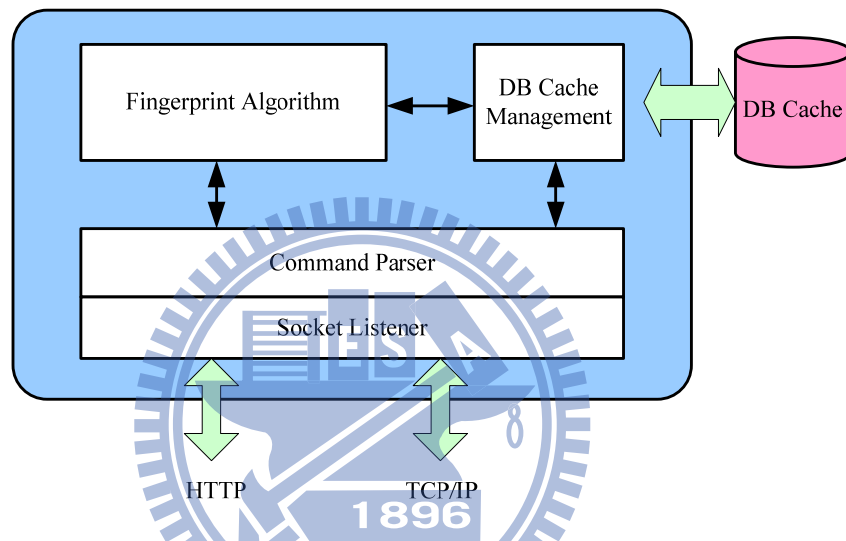


Figure 10. Fingerprint Server architecture.

3.1.2 Database Server

We adopt open source database MySQL (5.5.21) as Database Server in HFIS. TCP/IP Server and Web Server can access Database Server by JDBC (Java Data Base Connectivity) which is the Java API used for performing SQL command with regardless to the low level driver of database and the related accessing APIs.

Data stored in the database includes department information, entrance information, access log information, and user information such as user ID, user name, RFID, fingerprint data, and also the access authorization information. Besides, IT department can also create an

enterprise cloud space for internal resource access such as email system, on-line meeting, project status tracking, documentations sharing, and so on.

These applications are easy to use and very helpful to enhance the efficiency of enterprise management. In the meanwhile, because the industry properties protection is very important, using the fingerprint authentication to control the right of resource access and keeping access records for tracking can bring the higher security.

3.1.3 TCP/IP Server

TCP/IP Server developed by Java plays as a role of middle bridge among Embedded-based Client, Fingerprint Server and Database Server. It is always listening to the incoming messages on the specific port 1600 and responsible for data forwarding, flow control, and log management. It forwards all the request messages collected from Embedded-based Clients to Fingerprint Server and responds the execution result to the corresponding Embedded-based Client. Some of these execution results will be saved as the users' access records in the database.

TCP/IP Server, Fingerprint Server and Embedded-based Client are formed as a 3-tier client/server model and the back and forth messages are packed as TCP packets. The data packet between TCP/IP Server and Fingerprint Server is also the TCP/IP format.

3.1.4 Web Server

The Web Server is established by Apache (2.4.1) with PHP (5.4) development for data forwarding, flow control, and log management. Just like the TCP/IP Server, it is the middle and transparent bridge between PC-based Clients and Fingerprint Server as a 3-tier structure. It accepts the HTTP request messages from web browsers of PC-based Clients and forwards these requests to Fingerprint Server and replies the HTTP response messages back to clients.

3.1.5 Embedded-based Client

The Embedded-based Client we proposed is a lightweight fingerprint identification device (LwFD) which is developed based on ARM Cortex A8 SoC as the kernel chip with embedded 96 KB RAM, 256 KB flash and 802.3 MAC/PHY and the embedded Linux as the kernel system. The peripherals include the fingerprint reader, RFID reader, keypad and the LCD display. Figure 11 shows the hardware architecture of LwFD.

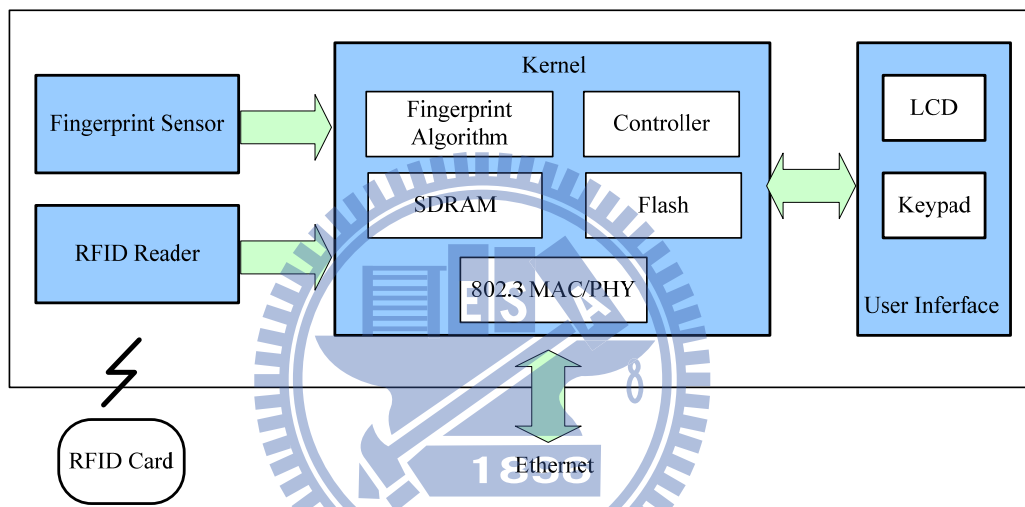


Figure 11. Hardware architecture of LwFD.

In contrast to PC-based Client, LwFD gets less computational power but much lower cost and lower power consumption. Beside the drivers of peripherals, the most important part of the kernel is the fingerprint algorithm porting. Fingerprint algorithm porting takes much computing power and large memory resources. The reason of fingerprint algorithm inside the Embedded-based Client is to extract the features from fingerprint images, but not for matching.

For double security mechanism, the feature can be transferred to server together with the RFID information [25] in encrypted format through TCP/IP protocols. The RFID is a secure card and is used as an index to perform one-to-one server matching which can reduce the

response time and also can be used as the backup solution for the situation of network failure. It means that depending on the users' access authorization, the RFID information of some group of users or all users should be stored in the embedded flash memory of the particular Embedded-based Client for the situation of network failure.

3.1.6 PC-based Client

With the HTTP protocols, the PC-based Client can be various platforms, such as Windows, Linux, iOS and Android on PC, laptop, tablet and even the smart phone. All these platforms connect to Web Server as the browser/server architecture. Each client can easily execute the web browser to access the web services provided by the Web Server. Figure 12 illustrates the Windows application design for PC-based Client.

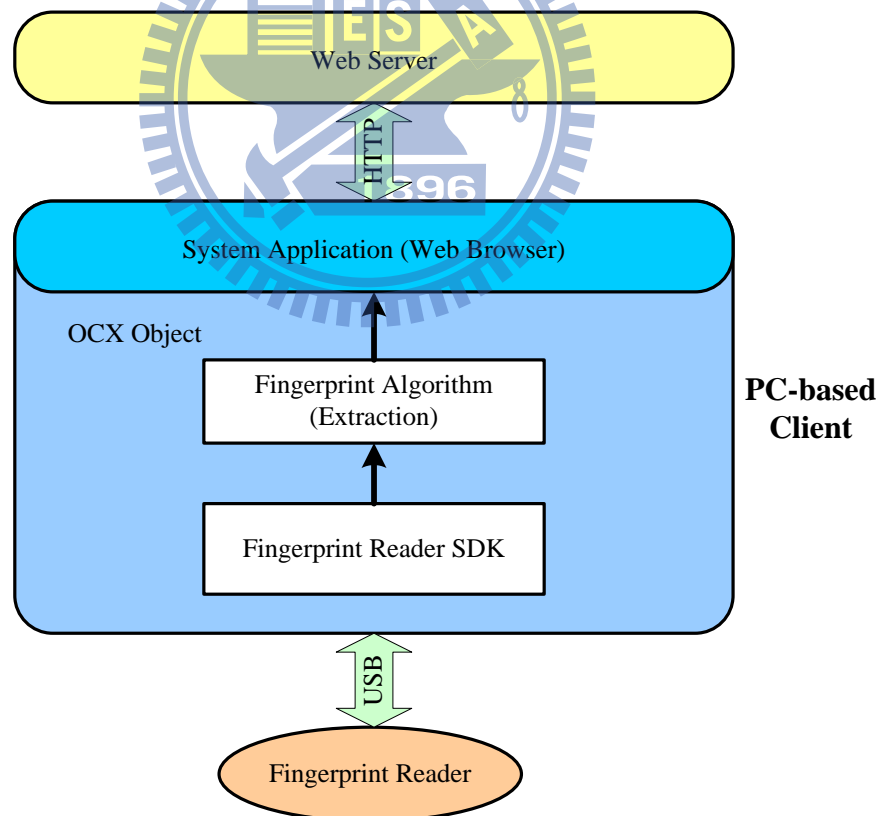


Figure 12. Windows application design for PC-based Client.

3.2 Traffic reduction

To consider the massive transactions of enterprise applications, such as time attendance and door access control, we design an efficient TCP/IP communication protocol to reduce the transmission data overhead which increases the overall system performance and reduces the system response time. Data transfer with HTTP protocol is always together with a set of headers. The headers will be a significant part when the size of actual data transferred is small. HTTP headers contain information about things such as last modified date, character encoding, server name and version and more. Figure 13 shows the packet of HFIS-HTTP in application layer has the extra overhead of HTTP headers compared to the packet of HFIS-TCP in transport layer.

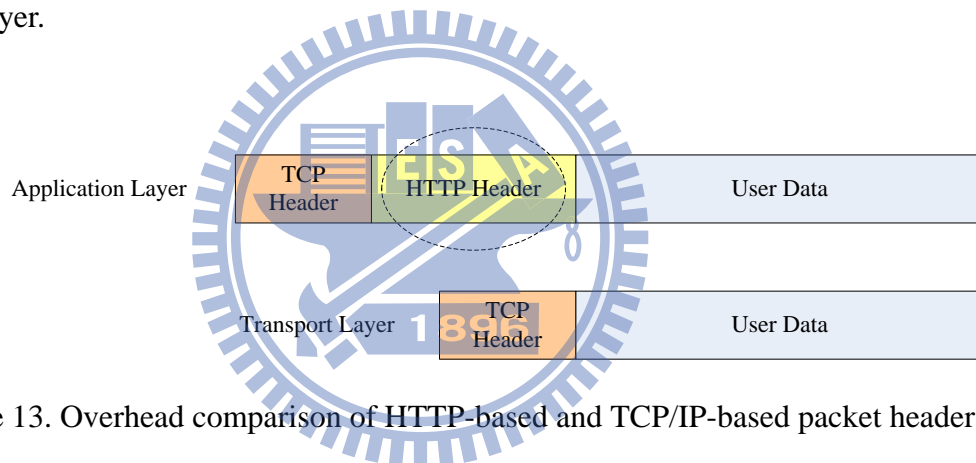


Figure 13. Overhead comparison of HTTP-based and TCP/IP-based packet headers.

Different web browsers and web servers have different size limitation for request URL. For browsers, the length limitation of URL in HTTP 1.1 is 2083 bytes for Microsoft IE browser and the length of other browsers can be longer. For the Apache server in our design, the maximum length of URL is 8192 bytes. Therefore, for HFIS-HTTP FIS, we proposed to transfer fingerprint feature in URL of request line instead of in the message body by XML can reduce the data size with no XML tags and declaration. That is, the fingerprint feature with the maximum size of 512 bytes generated from client is transferred in the URL of request line. Table 6 shows the HTTP request message for the fingerprint matching request with approx.

600 bytes of headers. Table 7 shows the HTTP response message has approx. 300 bytes of headers and the message body contains the matching result by XML format.

Table 6. HTTP request message (headers approx. 600 bytes).

HTTP Request	<i>Request Line</i>	GET http://211.22.110.7:8080/identify?template=XXX77HH HTTP/1.1
	<i>General Headers</i>	Connection: Keep-Alive
	<i>Request Headers</i>	Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, application/x-ms-application, application/x-ms-xbap, application/vnd.ms-xpsdocument, application/xaml+xml, */* Accept-Language: zh-tw Accept-Encoding: gzip, deflate User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; InfoPath.1; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; .NET CLR 1.1.4322; .NET4.0C) Host: 211.22.110.7:8080
	<i>Entity Headers</i>	
	<i>Message Body</i>	

Table 7. HTTP response message (headers approx. 300 bytes).

HTTP Response	<i>Status Line</i>	HTTP/1.1 200 OK
	<i>General Headers</i>	Date: Sat, 02 Jun 2012 12:50:03 GMT Connection: close Cache-Control: no-cache
	<i>Response Headers</i>	Server: YuServer/0.1.2
	<i>Entity Headers</i>	Content-Type: text/xml Content-Length: 153
	<i>Message Body</i>	<pre><?xml version="1.0" encoding="UTF-8"?> <yuserver sts="0" msg="OK"> <identify> <match score="128" uid="Test" fid="L1"/> </identify> </yuserver></pre>

The data transmitted from clients to server is the fingerprint features instead of the fingerprint images. Fingerprint feature is the extracted information from fingerprint image and its data size, maximum 512 bytes, is much smaller than the image size, 64 KB. There are some advantages to transmit fingerprint feature.

- Higher security
- Less data transmitted

Because features are extracted from a specific algorithm with specific parameters setting, the feature can be regarded as a kind of encrypted data. Transferring fingerprint image results in lots of TCP headers generated from data segmentation. So, the total transmission overhead of the HTTP headers and TCP headers can be reduced much by transferring features.

3.3 Offloading

System performance is evaluated by the response time which is highly related to:

$$\begin{aligned} \text{response time} = & \text{packet transmission time} + \\ & \text{propagation delay} + \\ & \text{queuing time} + \\ & \text{processing delay} \end{aligned}$$

To get a lower response time, we transfer fingerprint feature to reduce the packet transmission time. As to the processing delay, we can reduce it by offloading the task of feature extraction from Fingerprint Server to clients. As we know, the feature extraction is the most time-consuming task in fingerprint identification system. That means that the extraction in each client can reduce the overloading probability of Fingerprint Server and make the client requests wait less time in the processing queue of Fingerprint Server.

3.4 Features of our design approach

The main differences between the proposed HFIS and related work are described as follows. Shi [20] and Chang [22] adopt the Embedded-based Client and run the TCP/IP protocols for communication. Liu [19] and Wang [21] adopt the PC-based Client and the way of communication is through HTTP protocols. All of them transfer the fingerprint image to server for feature extraction and matching requests. However, HFIS integrates the Embedded-based FIS and PC-based FIS and proposes the hybrid system architecture. In HFIS, to enhance the system efficiency, we reduce the size of transmission data by transferring fingerprint features instead of the fingerprint images and the transmission performed in transport layer with TCP/IP protocol can even greatly decrease the protocol

overhead with no HTTP headers involved and no extra overhead of TCP headers coming from segmentation. Moreover, transmission of features can increase the data security and the feature extraction in each client brings the offloading effect.



Chapter 4

Evaluation and Discussion

4.1 Simulation setup

We use a system performance evaluation tool, HP LoadRunner 11.00 [15], to evaluate the proposed FIS architectures. By loading all user scripts simultaneously, we can simulate massive client requests occurred concurrently. The user scripts are virtual users used by LoadRunner as a replacement of the real behavior of human users. Here, we regard a virtual user as a client. Table 8 shows the server environment setup.

Table 8. Server environment setup.

Item	Description
OS	Ubuntu Linux 2.4.21-37.ELsmp #1 SMP
CPU	Intel(R) Xeon(TM) MP CPU 2.00 GHz x 4
Memory	4GB
Application	FPListener JDK1.5 Max Thread = 32 FPMatcher JDK1.5 Max Thread = 32 DB Connection Pool Max Size = 16
Database	MySQL 5.5.21

In our simulation, the FPListener, FPMatcher and MySQL database are running on a single server to represent the separated TCP/IP Server (Web Server), Fingerprint Server and Database Server respectively. The FPListener has the capabilities of listening to sockets, parsing TCP and HTTP packets and forwarding the fingerprint data to the FPMatcher. The FPMatcher is in charge of fingerprint extraction and fingerprint matching. All features stored in the database are preload to the cache of the server. Note that we assume there is no extra

database access time needed. The parameters settings in the simulation are summarized in Table 9.

Table 9. Simulation parameters.

Parameter	Value
Fingerprint database	10,000 features
Fingerprint image size	64 KB
Fingerprint feature size	512 bytes
Scenarios	Number of client is from 100 to 1500 with intervals of 100
TCP matching request	16 bytes
TCP matching response	16 bytes
HTTP matching request	600 bytes
HTTP matching response	300 bytes
Bandwidth	10 Mbps

We also assume that the extraction time of Fingerprint Server is 0.2 second and the average one-to-many server matching time is 0.5 second and one-to-one matching time is 0.1 second with 100% matching hit rate. Extraction in PC-based Client only takes 0.2 second, but the extraction time of Embedded-based Client is assumed 0.5 second, which is longer than that of PC-based Client.

4.2 Simulation results and discussion

The good and acceptable web page response time is less than 2 seconds [16]. Therefore, we used two evaluation metrics, *deadline miss ratio* and *average response time*, defined as follows, to evaluate simulation results. Besides, we analyzed the influences of the protocol overhead and offloading for both HFIS-HTTP and HFIS-TCP.

- Deadline miss ratio (DMR): the ratio of number of clients' requests that miss the response time threshold (e.g., two seconds)
- Average response time: the average time consumed from issuing matching requests until receiving responses of all clients under a specific deadline miss ratio (5%)

We generated scenarios to evaluate the deadline miss ratio for six FIS approaches, Liu [19], Wang [21], HFIS-HTTP with one-to-many server matching, HFIS-HTTP with one-to-one server matching, Chang [22], and HFIS-TCP with one-to-many server matching. These scenarios are of different number of clients from 100 to 1500 with intervals of 100. Figure 14 shows the deadline miss ratios of different FIS approaches.

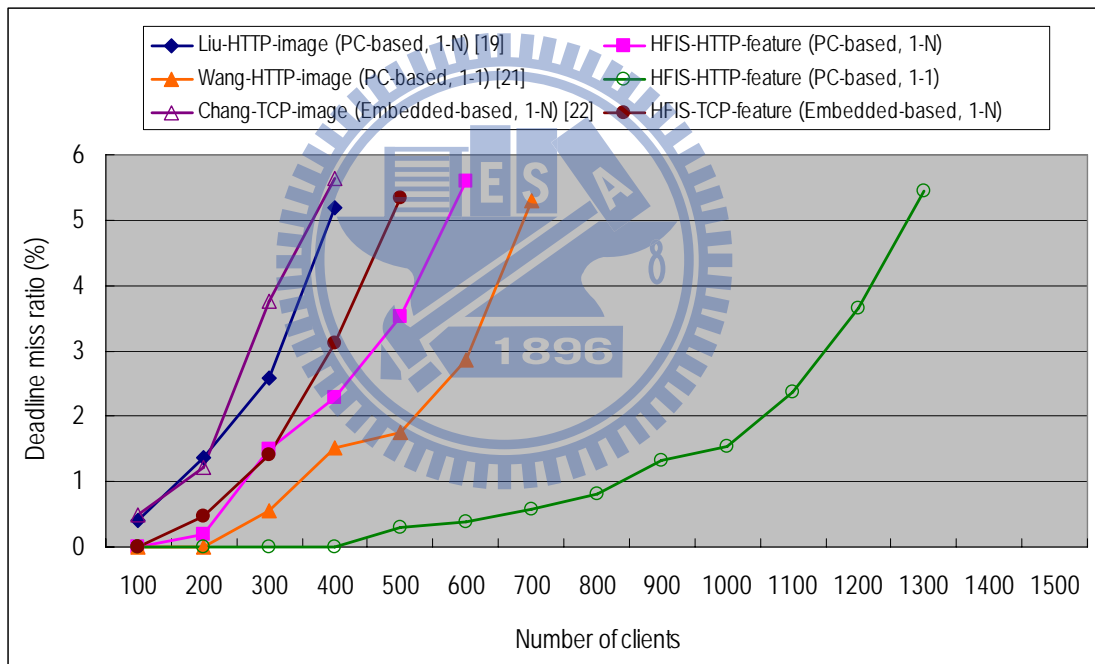


Figure 14. Deadline miss ratios of different FIS approaches.

As the number of clients grows, the deadline miss ratio increases. The deadline miss ratio of the proposed HFIS increases more slowly than that of related work with the number of clients increased. This is because when the number of client requests exceeds the limitation of concurrent threads that Fingerprint Server can provide, longer queuing time for service is needed. Liu-HTTP-image (PC-based, 1-N) [19] has to wait for feature extraction and

fingerprint matching in Fingerprint Server. But due to the feature extraction offloading to all clients, HFIS-HTTP-feature (PC-based, 1-N) only needs to wait for the time of fingerprint matching. That is why the HFIS-HTTP-feature (PC-based, 1-N) can support more number of clients than Liu-HTTP-image (PC-based, 1-N) [19]. Both Wang-HTTP-image (PC-based, 1-1) [21] and HFIS-HTTP-feature (PC-based, 1-1) perform the one-to-one server matching, so their increasing rate of deadline miss is much smaller than the one-to-many server matching of Liu-HTTP-image (PC-based, 1-N) [19] and HFIS-HTTP-feature (PC-based, 1-N). For the same reason depicted above, HFIS-HTTP-feature (PC-based, 1-1) can support more number of clients than Wang-HTTP-image (PC-based, 1-1) [21] and HFIS-TCP-feature (Embedded-based, 1-N) can support more number of clients than Chang-TCP-image (Embedded-based, 1-N) [22]. Table 10 illustrates the extra numbers of clients supported of HFIS-HTTP and HFIS-TCP compared to related work.

Table 10. Extra number of clients supported.

Deadline miss ratio Approach	0%	1%	2%	3%	4%	5%
	Liu-HTTP-image (PC-based, 1-N) [19]	28	74	109	144	165
HFIS-HTTP-feature (PC-based, 1-N)	(38.89%)	(40.66%)	(41.92%)	(44.86%)	(46.48%)	(48.83%)
Wang-HTTP-image (PC-based, 1-1) [21]	203	504	525	529	571	580
HFIS-HTTP-feature (PC-based, 1-1)	(103.05%)	(147.37%)	(99.62%)	(85.6%)	(87.98%)	(84.06%)
Chang-TCP-image (Embedded-base, 1-N) [22]	23	62	99	120	125	109
HFIS-TCP-feature (Embedded-based, 1-N)	(29.87%)	(32.8%)	(41.42%)	(45.28%)	(40.32%)	(30.19%)

We can get the number of clients supported and record the response time of each transaction under different deadline miss ratios for each approach. So, we calculated the average response time of different deadline miss ratio for each approach as shown in Figure 15. The average response time increases when deadline miss ratio increases. The proposed HFIS has shorter average response time than related work for all cases, except HFIS-TCP with Embedded-based Clients at lower deadline miss ratio. Compared to Chang-TCP-image (Embedded-based, 1-N) [22], HFIS-TCP-feature (Embedded-based, 1-N) has shorter average response time when the deadline miss ratio is not less than 2%; however, it has longer average response time at lower deadline miss ratios (0% and 1%). This is because our proposed approach assigns the task of feature extraction to each client, and the time consumed of feature extraction is highly related to the client capability. Nevertheless, it gets shorter average response time than Chang-TCP-image (embedded-based, 1-N) [22] with the deadline miss ratio increased.

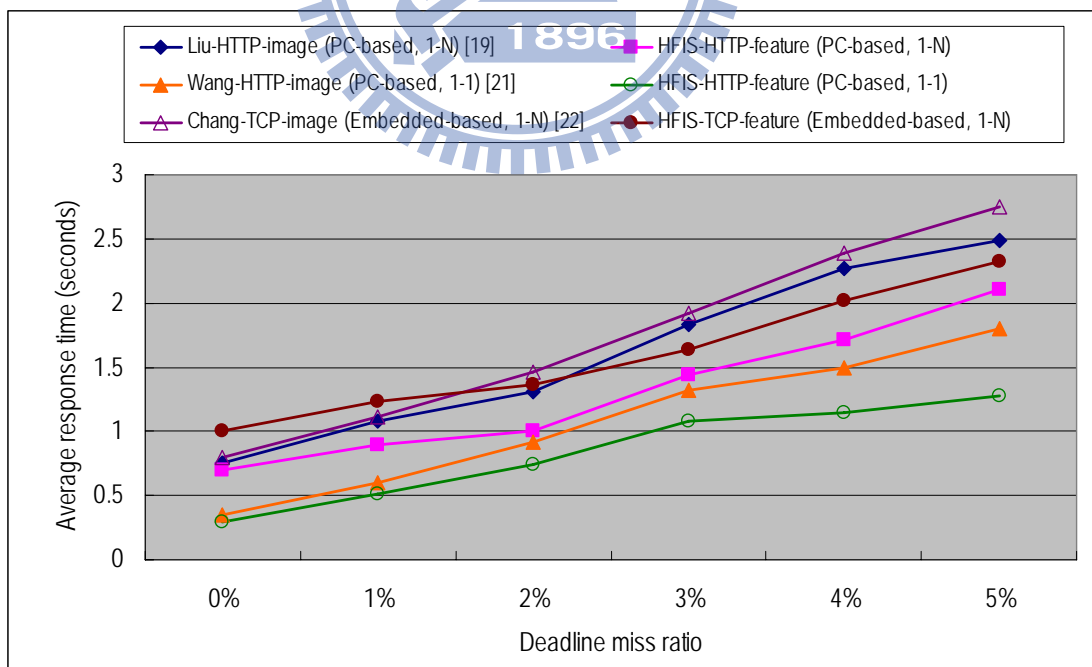


Figure 15. Average response time under different deadline miss ratios.

Table 11 shows the improvements of the average response time for the proposed HFIS compared to related work. We focus on results that have average response time under 2 seconds and found our proposed HFIS has better improvement at different deadline miss ratio: HFIS-HTTP-feature (PC-based, 1-N) at 2% DMR, HFIS-HTTP-feature (PC-based, 1-1) at 5% DMR, and HFIS-TCP-feature (Embedded-based, 1-N) at 3% DMR.

Table 11. Improvement of the average response time.

Deadline miss ratio	0%	1%	2%	3%	4%	5%
Approach						
Liu-HTTP-image (PC-based, 1-N) [19]	6.67%	16.67%	23.66%	21.31%	24.67%	15.66%
HFIS-HTTP-feature (PC-based, 1-N)						
Wang-HTTP-image (PC-based, 1-1) [21]	14.29%	15%	19.57%	18.18%	22.82%	28.89%
HFIS-HTTP-feature (PC-based, 1-1)						
Chang-TCP-image (Embedded-base, 1-N) [22]	-25%	-10.18%	6.85%	14.58%	14.14%	23.14%
HFIS-TCP-feature (Embedded-based, 1-N)						

Based on the assumption of 10 Mbps bandwidth, the data transmission time of feature transfer is decreased by 3.52% compared to image transfer. Compared to HFIS-HTTP, the traffic reduction ratio of HFIS-TCP is 52.08% and the protocol overhead greatly reduced from 55.56% (HFIS-HTTP) to 7.25% (HFIS-TCP). Less protocol overhead can prevent the waste of bandwidth in the network and increase the network efficiency. As to the offloading effect, it enhanced the system scalability with no doubt as our preceding discussion.

Chapter 5

Conclusion

5.1 Concluding remarks

In this thesis, we have presented a hybrid fingerprint identification system, HFIS, for both types of HTTP-based and TCP-based applications. The fingerprint feature is extracted at the client side and is then transferred to the server for matching. The transferred data is a feature instead of an image, so the feature itself presents as a kind of encryption and it can enhance data security. Compared to other image-based approaches, under 2 seconds response time threshold, simulation results have shown that the proposed HFIS-HTTP increases the number of clients supported by 41.92% (reduces average response time by 23.66%) compared to Liu's at 2% deadline miss ratio and by 84.06% (reduces average response time by 28.89%) compared to Wang's at 5% deadline miss ratio; HFIS-TCP increases the number of clients supported by 45.28% (reduces average response time by 14.58%) compared to Chang's at 3% deadline miss ratio. In summary, the proposed HFIS-HTTP and HFIS-TCP have shorter average response time than related work for all cases, except HFIS-TCP with Embedded-based Clients at lower deadline miss ratios (0% and 1%), and thus is very feasible for various enterprise applications which require different combinations of high security, low cost, low response time, and high scalability.

5.2 Future work

To support a large scale fingerprint identification system, we can adopt multiple servers and a proper load balancing policy to avoid a potential bottleneck in TCP/IP Server or Web

Server of the HFIS system. In addition, we may deploy the HFIS to a cloud computing environment with on-demand virtual servers to provide scalable fingerprint identification services according to the incoming request arrival rate. That is, designing a cloud-based fingerprint identification system is feasible for large enterprise applications that require high security and high scalability.



References

- [1] K. Uchida, "Fingerprint Identification," *NEC Journal of Advanced Technology*, vol. 2, No.1, pp. 19-27, January 2005.
- [2] T. Putte and J. Keuning, "Biometrical Fingerprint Recognition: Don't get your fingers burned," *IFIP TC8/WG8.8 Fourth Working Conference on Smart Card Research and Advanced Applications*, pp. 289-303, September 2000.
- [3] A.K. Jain, L. Hong, and R. Bolle, "On-Line Fingerprint Verification", *IEEE Transactions on PAMI*, pp. 302-314, 1997.
- [4] S. Ribaric, D. Ribaric, and N. Pavesic, "Multimodal biometric user-identification system for network-based applications," in *Proceedings of IEE Vision on Image and Signal Processing*, vol. 150, pp. 409-416, Dec. 2003.
- [5] P. Schaumont, D. Hwang, S. Yang, and I. Verbauwhede, "Multilevel Design Validation in a Secure Embedded System," *IEEE Transactions on Computers*, vol. 55, pp. 1380-1390, November 2006.
- [6] P. Schaumont, D. Hwang, and I. Verbauwhede, "Platform-Based Design for an Embedded-Fingerprint-Authentication Device," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, pp. 1929-1936, Dec. 2005.
- [7] V.K. Sagar, C. Greening, W.Y. Tan, and C.S.A. Leung, "Hardware/Software Co-Design of a Fingerprint Recognition System," *IEE Colloquium on Partitioning in Hardware-Software Codesigns*, pp. 1-5, Feb. 1995.
- [8] D.D. Hwang, and I. Verbauwhede, "Design of Portable Biometric Authenticators – Energy, Performance, and Security Tradeoffs," *IEEE Transactions on Consumer Electronics*, pp. 1222-1231, Nov. 2004.

- [9] M. Faundez-Zanuy., “A Door-Opening System Using a Low-Cost Fingerprint Scanner and a PC,” *IEEE on Aerospace and Electronic Systems Magazine*, pp. 23-26, Aug. 2004.
- [10] P.J. Phillips, A. Martin, C.L. Wilson, and M. Przybocki, “An Introduction Evaluating Biometric Systems,” *IEEE Computer*, pp. 56-63, Feb. 2000.
- [11] P.Y. Chan, and J.D. Enderle, “Automatic Door Opener,” in *Proceedings of the IEEE Bioengineering Conference*, pp. 139-140, April 2000.
- [12] P. Gupta, S. Ravi, A. Raghunathan, and N.K. Jha, “Efficient Fingerprint-based User Authentication for Embedded Systems,” in *Proceedings of Design Automation Conference*, pp. 244-247, June 2005.
- [13] G.C. Chao, S.S. Lee, H.C. Lai, and S.J. Horng, “Embedded Fingerprint Verification System,” in *Proceedings of 11th International Conference on Parallel and Distributed Systems*, pp. 52-56, July 2005.
- [14] S.J. Alotaibi, and D. Argles, “FingerID: A new security model based on fingerprint recognition for distributed systems,” in *Proceedings of 2011 World Congress on Internet Security (WorldCIS)*, pp. 284-289, 2011.
- [15] HP LoadRunner Software. Available:
<http://www8.hp.com/us/en/software-solutions/software.html?compURI=1175451#.T97OnlLdHhc>.
- [16] F. Fui and H. Nah, “A study on tolerable waiting time: how long are Web user willing to wait,” *Behavior and Information Technology*, vol. 23, issue 3, pp. 153-163, 2004.
- [17] D. Maltoni, D. Maio, A.K. Jain, and S. Prabhakar, “Handbook of Fingerprint Recognition,” 2nd Edition, 2009.
- [18] J. Goldie, “The Ways to Bulletproof RS-485 Interfaces,” *National Semiconductor Application Note 1057*, 1996.

- [19] W. Liu, C. Zhou, and Z. Ye, "Fingerprint Based Identity Authentication for Online Examination System," in *Proceedings of 2010 Second International Workshop on Education Technology and Computer Science*, vol. 3, pp. 307-310, 2010.
- [20] B. Shi, F. Xu, and M. Dai, "Design and implementation of wireless fingerprint identity authentication based on GPRS," in *Proceedings of 2010 Second International Conference Communication Systems on Networks and Applications*, vol. 1, pp. 286-289, 2010.
- [21] X. Wang, H. Zhang, T. Zhang, and L. Tong, "Design and Implementation of a Fingerprint Authentication System under B/S Architecture," in *Proceedings of 2009 CiSE on Computational Intelligence and Software Engineering*, pp. 1-4, 2009.
- [22] B.R. Chang, H.F. Tsai, C.M. Chen, and C.F. Huang, "Access Control of Cloud Computing Using Rapid Face and Fingerprint Identification," in *Proceedings of 2011 Second International Conference on Innovations in Bio-inspired Computing and Applications*, pp. 179-182, 2011.
- [23] Q. Shafi, J. Khan, N. Munir, and N.K. Baloch, "Fingerprint verification over the network and its application in attendance management," in *Proceedings of 2010 International Conference on Electronics and Information Engineering (ICEIE)*, vol. 2, pp. 555-559, 2010.
- [24] J. Li, X. Zhu, X. Li, Z. Zhang, and J. Sui, "Wireless Fingerprint Attendance System Based on ZigBee Technology," in *Proceedings of 2010 2nd International Workshop on Intelligent Systems and Applications (ISA)*, pp. 1-4, 2010.
- [25] B. Fernandez-Saavedra, R. Alonso-Moreno, A. Mendaza-Ormaza, and R. Sanchez-Reillo, "Usability Evaluation of Fingerprint Based Access Control Systems," in *Proceedings of 2010 6th International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp. 333-336, 2010.