



Czech Technical University in Prague

Faculty of Transportation Sciences

Aliaksandr Kuchun

Design of biometric applet for CTU ID card

Master's Thesis

2016



K614..... Department of Applied Informatics in Transportation

MASTER'S THESIS ASSIGNMENT

(PROJECT, WORK OF ART)

Student's name and surname (including degrees):

Bc. Aliaksandr Kuchun

Code of study programme code and study field of the student:

N 3710 – IS – Intelligent Transport Systems

Theme title (in Czech): **Návrh biometrického appletu pro ID kartu ČVUT**

Theme title (in English): Biometric applet design for CTU ID card

Guides for elaboration

During the elaboration of the master's thesis follow the outline below:

- Biometric methods analysis
- Selected method description
- CTU ID card - technical solution design
- System architecture design
- Biometric applet programming
- Solution testing

Graphical work range: according to supervisor's recommendations

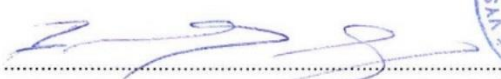
Accompanying report length: min. 55 pages including figures, graphs and tables


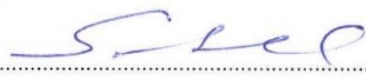
Bibliography: Ashbourn, J.: Practical Biometrics - From Aspiration to Implementation, Springer Verlag, 2004, ISBN 1-85233-774-5
Bhanu, B., Tan, X.: Computational Algorithms for Fingerprint Recognition, Kluwer Academic Publishers, USA, 2004, ISBN 1-4020-7651-7

Master's thesis supervisor: **Ing. Jana Kaliková, Ph.D.**

Date of master's thesis assignment: **July 30, 2015**
(date of the first assignment of this work, that has be minimum of 10 months before the deadline of the theses submission based on the standard duration of the study)

Date of master's thesis submission: **November 30, 2016**
a) date of first anticipated submission of the thesis based on the standard study duration and the recommended study time schedule
b) in case of postponing the submission of the thesis, next submission date results from the recommended time schedule


doc. Dr. Ing. Tomáš Brandejský
head of the Department
of Applied Informatics in Transportation


L.S.

prof. Dr. Ing. Miroslav Svítek, dr. h. c.
dean of the faculty

I confirm assumption of master's thesis assignment.


Bc. Aliaksandr Kuchun
Student's name and signature

PragueJune 6, 2016

Acknowledgement

I would first like to thank my thesis advisor Ph.D. Jana Kaliková from Department of Applied Informatics in Transportation, for her help during the whole project. I would also like to thank Ing. Radek for his assistance during creation of the practical part. Finally, I would also like to acknowledge my relatives and friends, who supported me during my master studies.

Declaration

I hereby, declare that this thesis is my own work and that, to the best of my knowledge and belief, it contains no material which has been accepted or submitted for the award of and other degree or diploma.

I also declare that, to the best of my knowledge and belief, this thesis contains no material previously published or written by any other person except where due reference is made in the text of the thesis.

Prague, October 28th, 2014

Aliaksandr Kuchun

Abstrakt

Cílem diplomové práce "Návrh biometrického appletu pro kartu ČVUT" je analýza potenciálního využití biometrie v ČVUT, vývoj optimální architektury systému podle aktuálního stavu stávajícího systému a zavedení kódu appletu, který zajišťuje funkčnost kombinované RFID a biometrické čtečky.

Klíčová slova

Biometrie, biometrické metody, identifikace, verifikace, otisk prstu, systém, systémové inženýrství, RFID, Mifare, DESfire, APDU

Abstract

The aim of the master's thesis "Design of biometric applet for CTU card" is analysis of the potential usage of biometrics in CTU, development of optimal system architecture according to the current system's state and introduction of applet code, which ensures functionality of combined RFID and biometric reader.

Key words

Biometrics, biometric methods, identification, verification, fingerprint, system, system engineering, RFID, Mifare, DESfire, APDU

Contents

1. Introduction to biometrics	4
1.1 General view on identity and identification	4
1.2 Person identification	5
1.3 Technological and informational aspects of identification.....	8
1.4 Biometric identification and verification	10
2. Method selection.....	14
2.1 General project specification	14
2.2 Selection of biometric method and equipment	16
2.3 Technical and processual description of selected method	19
3. System proposal	24
3.1 System architecture.....	24
3.2 RFID.....	26
3.2.1 Technology description	26
3.2.2 RFID system at CTU	27
3.2.3 Instructions language	28
3.2.4 Reader provided	29
3.3 System analysis	29
3.3.1 System elements and their functions.....	29
3.3.2 System's competence	31
3.3.3 Activity diagram.....	31
3.3.4 System requirements	31
3.4 Software Development Kit (SDK).....	32
3.4.1 General Description	32
3.4.2 NPX APDU	33
3.4.3 MifareGlobal library	35
3.4.4 Documentation.....	35
3.5 Program	36
3.5.1 Development issues and limitations	36
3.5.2 Language selection and pre-requests	37
3.5.3 Code functionality	39
3.5.4 Code implementation	41
3.5.5 Notes for the code.....	45
4. Results.....	55

4.1 Final proposal analysis	55
4.2 Comparison with the alternatives	56
4.3 Usage of the project	58
4.4 Conclusion.....	59
List of attachments.....	60
List of abbreviations	61
List of figures	62
List of tables.....	63
Literature and references	64

1. Introduction to biometrics

1.1 General view on identity and identification

An object's *identity* could be defined as “necessary requirements, which need to be fulfilled in order to be someone or something”. Human identity consists of biological and mental, congenital and obtained, individual and social properties. A human can have several identities during his life - not only criminals, but, for example, actors, special forces members, agents. Despite such a variety of identity components, the interest of the thesis lies in its biological aspects. The biological identity of a person is the combination of its congenital and obtained individual biological properties, which are independent on human's will. Object's identity can consist of various parameters, such as unique code, weight, size, etc. It mostly depends on the type of the object and the purpose of identification.

The process of *identification* is usually described as a “*process of detection or proving identity*”. Basically it is comparison of different objects based on their similarities in properties, form, structure, functions, etc. in order to detect if those objects are similar or not. This is a decisive process, which might have a very variable nature and structure. It might be said that this criterion is also rigorous, as some of the properties of the object might change in time: for example, human's appearance might be affected by age.

The ability to identify a person using its unique attributes (like name, age, nationality, etc.) is crucial within the society. Human's body characteristics, such as face, voice, together with other contextual information, like clothes, location, are being used by people to identify each other. This set of the attributes is a part of person's identity. Nowadays, due to the mobility of the modern people, society growth and other different factors, modern man faces lots of unknown people, objects, events, which could even be on the other side of the planet, have necessitated the development of modern identity management systems, which can efficiently record, maintain and obliterate individual identities.

Those systems play critical role in a number of industries, and their number continues to grow year by year. As an example a border crossing control, physical (nuclear plants, military zones) and logical (information) access restriction systems might be used. Identification applications could be separated into six categories: information registration, identification and information search, identification and object search, management and control of state, human identification and tracking, transaction processes. Information registration is an

application, which does not directly influence the main process and is used only for collecting data - for example, attendance systems. Identification and information search applications also have no direct consequences, and are used primarily for retrieving an information using identification symbol, like retrieving information about the goods via barcode. Object search, on the other hand, intends retrieving the associated object (ex. picking up order from a courier). Management and control of state applications are used for controlling the process associated with the object, for example in logistics when tracking the parcel. The most significant, and probably the most demanding area is human identification and tracking, which will be described particularly later. Transaction processes mainly include different bank and financial operations. The most commonly used technology nowadays is RFID.

1.2 Person identification

But what about the person identification? It is a very specialized and specific case. Basically it is said that there are two types of person identification: inner (self-identification) and outer. The inner identification is how a person sees itself, all its aspects (including psychological, social, etc.). The outer identification, on the other hand, is a summary of all technological methods, which could be used to identify a person via the set of the characteristics.

There are different points of view on the issue of person identification, and a wide range of technologies and methods currently used. We may identify a person by its behaviour, name, social role, physical, biological, security characteristics. Or, for example, via obtained disease, injuries or surgical intervention consequences, which is often used in medicine and criminalistics. In this case, it is very important to separate temporary (such as scratches, cuts) and permanent changes. Nevertheless, all person identification processes might be separated into 3 sections: *ownership* (holding a key, card, etc.), *knowledge* (password, PIN code, ID, etc.) and *biometric characteristics* (fingerprint scan, eye scan, voice identification, signature, etc.).

The first one, *ownership*, is currently the most wide-used person identification method. It is used in private, commercial and governmental sector for authorizing the legal person to the communities, objects, technical equipment, locations, finances, information, etc. Those restrictions have different nature: administrative, security, economical. The process of identification may be either forced or voluntary. The equipment used by those methods are personal documents, ID cards, microchips, and are most often combined with each other or another identification method.

Nevertheless, reliability of such methodology is a good question, especially in case of non-personal (automatized) process, even not taking into consideration the probability of key/card loss. Using only the names, for example, might cause a big problem as, at first, depending on the nationality or locality the person comes from, the names may have different pre- and postfixes, middle names, etc. (*al, ibn* in Middle East, *de* in France, second names in United States); second, the names may be transcribed from the other language differently (German Müller might easily turn into Muller); third, some of the nationalities have high frequency of occurrence of most characteristic names (Kim on Korean peninsula), and might lead to false positive identification. There are also other reasons, like, for example, second name change due to marriage and inability to identify person's sex from the name. Needless to say that the name itself does not grant a security level enough for almost any case, except from those, where the security itself is not required.

The second section is *knowledge*. This is probably the most used methodology group used nowadays in the Internet, and respectively the one showing the fastest popularity growth during the last years. This section's main and probably the most recognized element is password. It is now used on almost every website, providing protection of personal data, finances, documents, etc. Commonly it is used in form of "login/email/etc. + password".

This type of protection, nevertheless, has its own disadvantages. Theoretically it provides a strong security level, as the only place, where the password has to be kept is user's mind. Practically, however, this is a terrible place to keep such vulnerable information, as most of the people cannot remember strong passwords. They either write them down somewhere or end up using the same password for every system. Both steps significantly reduce the security level provided.

Weak passwords, on the other hand, are comparatively easy to crack. Taking into consideration the insecurity of the modern Internet, where viruses, which are able to catch your password and send it to a hacker, could be masked via browser plugin, it is obvious that the password systems are not reliable enough for being used in such systems. That is why they are supported by additional supportive systems, such as second password, SMS authorisation, etc. For example, financial transaction systems, such as online-banking, due to decentralisation of the customer service have led to the risk of identity theft, which made them to add additional protection layers. We might say that the most common source of mistakes leading to password crack are, mainly, people themselves. Such methods as social engineering, for example, would not have worked without human factor. Even young people, getting on easily with modern PCs, smartphones, etc., often become victims because of

installing fake apps or authorizing through untrusted services. So what can we say about older people, who might easily get caught by fake online banking service, enter their bank account data? That is why passwords cannot be potentially seen as trusted protection source.

The third section is *biometric characteristics*. The history of the method has begun in Ancient Egypt. There is an evidence of centralized measurements of peasant's height, weight, age, wounds and skin colour, which were documented and stored. Those data were later used in order to avoid multiple labour payment for the same person. In Ancient China, Persia and Babylon used dactyloscopy for authentication (for example, signing the document). Starting from 17th century, dactyloscopy became an object of interest for different scientists, such as M.Malpighi and J.E.Purkyně. Usage of dactyloscopy in criminalistics was popularized by W.J.Herschel, H.Faulds, F.Galton and E.Henry, J.Vucetich.[1] Interesting fact is that all their activities have taken place in foreign countries (India, Japan, Argentina). As a result, dactyloscopy became a commonly used method in criminalistics at the end of 19th century. In 1924 US Congress stated dactylography to be the primarily used identification method by FBI. In 1971 the database contained around 200 million of fingerprints. The paper evidence was ended in 1999, and the further process was implemented by AFIS (Automated Fingerprint Information System). The other system needed to be mentioned is hand geometry measurement system, which was used on Wall Street from 1970 to 1980 for authentication and attendance control.

Up to middle of 20th century, biometric identification was used solely in criminalistics and judgement processing. Nowadays it might be used in all aspects of human's life. The main advantages of those methods are:

- inability of key loss (unlike card or password)
- is hard or sometimes even impossible to steal or falsify the key
- is non-transferable
- high precision and identification speed
- easy to use
- is natural in context of human's essence
- possibility of partial or full automation

The application areas of biometric identification are:

- *Transaction security*: due to growing popularity of bank cards usage (and, respectively, number of attempts to use it for criminal acts) a new generation of ATM, using biometric data attached to the card, is already developed
- *Entrance security*: different objects (private and public; commercial and governmental) require different entrance restrictions. Attempts to avoid them, however, occur more and more often. That is why biometric identification is the most appropriate solution for this problem due to high level of security.
- *Travel and tourism*: from simple reservation systems, up to border control and migration flow control, which is nowadays one of the most demanding problem in Europe, identification processes require complex solutions, which only biometric identification can offer
- *Customer Relationships Management*: fast and reliable customer identification can obviously help to improve Business Processes
- *Property Security*: some types of property, for example luxury cars, require additional security restrictions, as they are often a tempting object for different types of car hijackers, burglars, etc. biometric Identification Systems can dramatically decrease the number of successive theft attempts.
- *Telecommunication security*: modern telecommunications also have a demand in high security level, as they are used for online transactions (e-commerce, e-banking) and transfer of sensitive data.
- *Attendance control*
- *Criminalistics*

It is obvious that already now the usage of biometric identification is expanding in many areas, and in the nearest future the tempo will not decrease.

1.3 Technological and informational aspects of identification

Identification has been used before solely in security domain. Large amounts of information required of police and secret services to integrate automation in gathering procedures in order to increase the procession speed and quality. Those projects were well invested and highly secured. The investigations were concentrated on all the forms of information. Practically those forms often interweave with each other. Nevertheless, in theoretical meaning, they are considered separately.

Pictorial information: a great accent has been put on pictorial information as of its strong informativity, its simple visualisation and understanding. Those were not only simple photographs, but also results of thermal, magnetic and radar analysis. This combination can bring users a unique information: for example, presence of living beings, or aircraft presence. Also a newer area of science called “computer vision” has been developed. It was dealing with problematics of digitizing images, filtering, pattern recognition and developing different comparison algorithms, which resulted in possibilities of objects detection and identification.

Textual information: in this case, the most commonly used method is so called full text technology. It is used for finding the keywords in different electronic information sources (databases, libraries, news, etc.). Such method allows to achieve the goal effectively and quick. In case of handwritten text, the task becomes more complex as we need to applicate pictorial information gathering first to get the digital version of the document. Nevertheless, handwritten document contains much more information about the author, the consequences and etc., which is often used by criminalists.

Acoustic information: here the process depends on the input: it might be either nature sounds, of which we may try to get information about the surrounding; vehicle sound, which is used to gather information about vehicle type, its engine, movement direction; human voice. Sometimes methods from visual information gathering are used, as far as the sound might be transformed to graphical form. Combination with full text technology also brings important improvements, allowing security services, for example, identify the voice records containing certain keywords.

Electronical information: this type is combined with any of the previous types, as most information nowadays is stored or transferred in electronic form. Information is gathered by monitoring all electromagnetic waves or signals from any source in the area of interest. The sharpest technologies for such processes are not publically available.

The main issue, which security services face now, is globalisation. Technologies, which could couple of decades ago be used only by a number of institutions, are now available for almost everyone. That’s why being a step ahead of public technologies is the primary target for all the security services around the world, though the price of such leadership might sometimes be very high.

The greatest change in the world of identification technologies was brought by computerisation: automation, quick search, etc. Nevertheless, it also brought lots of new

threats and vulnerabilities: the damage from identity theft are estimated around 8 bln. \$ annually.

1.4 Biometric identification and verification

The word “biometrics” is known for over 30 years already. Most of the society became familiar with it thanks Hollywood, for example, “Mission Impossible” movie series. That is why it has been often threaten as nothing but producer’s fantasies. Nevertheless, all those devices were somehow inspired by existing systems, which were hidden from public for a long time for security purposes. Massive spread of those technologies started with the commercial usage.

The expression itself means a summary of all the scientific knowledge, based mainly on statistical and analytical approach, which studies retrieving, processing and afterwards usage of measurable parameters of living beings aiming its unique identification and verification.

Biometric parameters are separated into two main groups: physiological and behavioural. The first group are the constant parameters, which do not depend on person’s behaviour, and could be summarized as certain parameters of human’s body parts, which appear to be unique for every person. The second one are products of certain human actions, which theoretically prove to be unique in most cases.

Physiological biometric attributes: [2]

- Iris
- Retina
- Face
- Ear shape
- Fingerprint
- Hand geometry
- Wrist vein topography
- Body smell
- DNA
- Wage and height
- Salt concentration

Nevertheless, some of them could be modified by the person, some others change constantly during the whole life (which makes them almost useless for practical application). The only one truly immutable parameter is DNA, which thought is not studied well enough to find massive application, as the technology level does not afford to do it fast and cheap enough.

Behavioural biometric attributes:

- Voice
- Locomotion
- Handwriting
- Signature
- Keyboard writing dynamics

Those are practically less used, as a trained person may change them the way it wants.

Assumption for usage of any characteristic in biometrics is its uniqueness, constancy, practical scalability and possibility of further technological processing aimed on comparison of values of this characteristic, which belong to different individuals.

Modern biometrics is inevitable connected with computer technologies. Automation of the identification process is a prior attribute of biometric identification. Definition “computer vision”, which includes search, detection and comparison of images, animation and objects, is now strongly connected to identification in different industrial sectors, and is often used in biometrics too. Other studies work with voice recognition, robotics and artificial intelligence.

Separately stands in this case DNA recognition. This is 100% reliable biometric identification method. Currently it is not automatized due to lack of technology, but hopefully in the near future it will be possible to make it quick and cheap. The same path was once completed by fingerprint recognition, which is now a commonly used method.

From the processing point of view, we may classify biometric methods into three groups: criminalistics, commercial and esoteric. The differences between those groups lie in all aspects of the process, requirements and results evaluation. criminalistics identification has much stricter detection ability criterion ($1:10^7$ up to $1:10^9$ vs $1:10^4$ up to $1:10^6$ by commercial identification) [1], but doesn't necessarily require full process automation. Also identification is prior for criminalistics processes, while verification - for commercial ones. The error consequences are much more dramatic, as it may allow criminal to escape justice. The

database contains not only the data of people, directly involved into the process, but also unknown identities (whose biometric traces were, for example, found on the crime scene). Also, the processing speed does not have to take only a few seconds, as it is in commercial case, because DNA analysis, for example, takes several days to be done. criminalistics identification often uses non-public, expensive methods, and the requirements for equipment are usually much stricter.

Esoteric biometric identification is separate methods group, which includes practically not used, often not even tested methods. Many of them are still in theoretical development stage. All of them are inaccessible to the public, and are practically used nowhere, except from laboratories. Esoteric identification is on the sharp edge of science and technology, receiving great finances for the research, and having “top secret” stamp on the whole project.

As an example of solely criminalistics method could be named DNA analysis. Fingerprint, voice, handwriting and hand geometry recognition are used both in criminalistics and commercial sphere. Esoteric identification includes, for example, locomotion, ear shape, body smell, etc.

Now let's have a brief view on the commonly used recognition methods.

- *Fingerprint*: the most popular method, “symbol” of biometrics, is a worldwide recognized standard for more than 100 years, for both commercial and criminalistics usage. This method is based on unique images of papillary lines. Nowadays, the images are taken not only with classical dactyloscopy, but using modern optoelectronic sensors.
- *Iris*: the colourful circle around the eye pupil contains unique identification points, using which enables to identify the person with an extremely high precision. Iris consists of randomly placed and constant in time colour structures. No two irises are identical. Capturing is implemented via standard video technology.
- *Retina*: The human retina is a thin tissue composed of neural cells that is located in the posterior portion of the eye. Because of the complex structure of the capillaries that supply the retina with blood, each person's retina is unique. The network of blood vessels in the retina is not entirely genetically determined and thus even identical twins do not share a similar pattern. Nevertheless, retina might change due to several diseases, such as diabetes and glaucoma. The capturing is done via IR scanner.
- *Hand geometry*: identifies users by the shape of their hands. Hand geometry readers measure a user's hand along many dimensions and compare those measurements to

measurements stored in a file. Viable hand geometry devices have been manufactured since the early 1980s, making hand geometry the first biometric to find widespread computerized use.

- *Voice*: human voice also contains unique behavioural and physiological characteristics. They are used in specific software tools, after the voice is transformed into unique digital code.
- *Face*: human face contains identification (anthropological) points, which are specific and immutable due to time.
- *Signature*: behavioural characteristic, which might be used for person identification. Not only the image of signature is processed, but also handwriting speed, pen pressure, writing direction, etc.
- *DNA*: potentially the most accurate and reliable method. Contains a huge amount of information about a person. Even small amount of it is enough for 100% recognition. Once will become the same, what fingerprint recognition is now.

2. Method selection

2.1 General project specification

As it was written in the previous section, security in modern world's context is required almost everywhere around - in our houses, at our jobs, in different governmental institutions, when we browse the Internet. With the growing technological level of the thefts, who might even be enemy countries' secret service units (which means they have an access to the top technologies and have advanced experience in tricking or disabling security systems), the demand in higher-levelled protection grows simultaneously.

Czech Technical University in Prague is also a place, which requires strong restrictions in some places because of the several reasons:

- Documents unauthorized operations: such activities, as stealing, copying, editing, throwing up important documents in the authorities' offices (such as dean, secretary, or the principal). Such activities might be theoretically, because of different reasons, done, for example, by students, competitors or companies. The consequences of such action may vary from comparatively insignificant up to very serious. Though the possibility of its occurrence is relatively low, it still has to be taken into account.
- Theft attempt: some of the classes and laboratories are equipped with machinery, sometimes costing millions of Czech Crowns (for example, diagnostic equipment at FBMI). Its value makes it a potential target for thieves, also because universities traditionally do not belong to the highly secured objects. Consequences might lead to great financial losses.

Security and restriction at CTU are currently provided by RFID cards, which are used by students, teachers and employees. This means that CTU has a certain security level, which nevertheless, has several issues. From the front-end communication, those are: unauthorized access to tags, rogue and clone tags and side-channel attacks.

Those threats are based on rogue devices, installed on the RFID readers, allowing them to read or modify tags from RFID card. Sometimes even full rogue readers, looking similarly to the real ones. Research carried out by different institutions revealed the weak points of RFID technology, such as RFID Virus, cell phone side channel attack and SpeedPass hack. And if

the probability of the first two cases in reality is comparatively low, the third one is a risky case.

Not only front-end communication brings holes in RFID's security concept. Tags are difficult to remove from the chip, which means that contents of the card often might be read after formal card's life cycle. Also, some high-gain antennas are able to read chip's content distantly, as the RFID technology doesn't support any authorisation - the card basically interconnects with any reader.

Human factor (in this case, card loss probability) should be also taken into account. This situation may give potential thieves a direct and guaranteed access to the target area without a necessity of using any hacking method.

All those potential threats bring up a question: what can we do with the situation in order to improve the security level? In order to answer it, we need to take into account several factors, which are:

- 1) Application area: the target is not to improve the security in general, as for most of the university rooms/classes/areas RFID is more than enough. Nevertheless, employee's rooms, laboratories with dangerous materials and expensive equipment should definitely have additional restrictions.
- 2) Access speed: the faster - the better, however no strict roof border should be applied.
- 3) Role restrictions: some users may have higher privileges than others.
- 4) Competence: there should be no (or at least almost no) way to either avoid or to trick the system for the average thieves
- 5) Confidentiality: the system specification should exclude the possibility of retrieving the access key.

The system already uses ownership pattern (RFID card in our case) for identification. Applying any method from the same group definitely will not significantly increase the security level of the whole system, which means that the method, which will be chosen as a complement, should belong to one of the other two groups - knowledge or biometric characteristics.

From the first group, the only optimal alternative is installing additional PIN-locks. In this case, each room will have its own 4-8 digits PIN code, entering which a person will receive an access to the room. Using password solution will require installing sensor displays, and is generally inappropriate solution in such case as of its input speed. Other knowledge-only

based methods are not implementable in this case, as they will require human control presence.

The PIN-based solution has the following advantages:

- Its implementation is technologically easier - installation and configuration of 10-digits numeric lock will not require much efforts.
- It is comparatively cheap - the equipment will not cost much
- It will require small time to implement - no centralized actions needed
- It is easy to maintain - in case the password needs to be changed, the action might be done directly through the lock

However, PIN-lock also has one significant drawbacks, which make its implementation almost useless: its security level increase is very low. It won't take much for an interested person to receive the password or to avoid this measure. There is a number of ways to get the password: fluorescent markers, social networking, installing camera or studying the lock in order to get the most threadbare buttons (which will significantly reduce the number of possible combinations, and make the brute force method applicable). Keeping a digital PIN in secret is an almost impossible task. The other factor, which needs to be taken into account is access speed: entering PIN code takes several seconds. Also, as human memory has its own drawbacks, it is possible that the new worker may forget one or more digits, so it will take him more time to enter the building.

So, this way we come to the third set of the methods - biometric characteristics. This set requires more deep studies, as it is not so widespread in this kind of institutions. Of course, saying "biometric characteristics" we do mean all of them, as long as some of them are simply unavailable to CTU either due to their price or due to legislative restrictions.

2.2 Selection of biometric method and equipment

Before performance measurements description of different methods, the criteria for the evaluation should be defined. Currently there are 5 criteria groups, which are clearly separated according to their target: *operational* (properties of the biometric parameter, such as uniqueness, immutability, etc.), *technical* (processing properties, mainly connected with the complexity and modernity of the equipment, ex. processing speed, effectivity, flexibility, etc.), *productive* (general properties of the system in context of industry), *financial* (not only equipment price, but also maintenance and staff training) and *methodological, security and*

algorithmic (complexity of algorithm, encoding, database, connections, etc.). All of them should be measured and taken into consideration before it is decided which one method will be used.

Also we need to define the method set we are interested in. While not all the methods are appropriate or available, the most common and affordable are chosen. Those are hand geometry, fingerprint, iris, retina, handwriting and voice recognition. Body smell, locomotion, ear shape recognition belongs to esoteric method, which makes them inapplicable by default. Wrist vein and face recognition, though do not belong to this group, are also not widespread and expensive, and the corresponding equipment is hard to buy. Instead, the finger vein scanners could have been used for this purpose. Other methods, such as weight/height measurements, cannot be tolerated as of their obsolescence - they have long processing time and low reliability.

Evaluation by operational criteria requires estimation of the requirements for the method in 3 fields: immutability, uniqueness and acceptability. Biometric attribute should not be varying, as it may lead to high percentage of false positive and false negative evaluations. According to the statistics, there are over 20000 people (including students, teachers, staff and external workers) at CTU. The satisfying floor uniqueness in this case is 1:100000, which will almost guarantee that no two similar values appear in university's area. Method also shouldn't have low acceptance level.

Now that the method set has been chosen, criteria are defined and requirements are determined, every one of those methods must be evaluated according to the operational criteria defined before. The results are available at Table 1.

Table 1: Operational criteria of biometric methods [2]

Method	Scanning	Immutability	Uniqueness	Acceptability
Hand geometry	Optical	Good	1:10000	Very good
Retina	Optical - laser	Very good	1:1000000	Bad (invasive)
Iris	Optical	Very good	1:6000000	Bad
Voice	Electroacoustic	Varying	1:10000	Good

Fingerprint	Optical, electronical	Very good	1:1000000	Good
Handwriting	Static or dynamic image	Varying	1:10000	Very good

From this table you can see that, despite all the methods have their advantages and drawbacks, the only method which corresponds with the requirements is fingerprint recognition. Hand geometry, voice and handwriting recognition have uniqueness, which is above the required. Moreover, the last two also provide low immutability level. The two eye-connected methods, iris and retina scan, have low acceptability due to the fact that, for example, people wearing contact lenses should remove them in order to pass the procedure. This may be threatened as discriminative restriction.

This way there is only one method, which meets the requirements for the system. Also:

- Fingerprint recognition is financially affordable nowadays, and costs significantly less than, for example, retina scanner
- Because fingerprint recognition is one of the most popular biometric methods currently used, market offers wide choice of different readers, which can be easily combined with other readers and systems through popular interfaces
- Algorithmic complexity in this case does not matter, as most of the readers have built in reading, encoding and templating functionality implemented

The additional comparison of 6 most popular methods can be found below, in Table 2

Table 2: General comparison of 6 most popular methods [3]

	Finger	Face	Voice	Iris	Hand	Signature
Maturity	very high	medium	medium	medium	high	medium
Sensor type	contact	non obtrusive	non obtrusive	non obtrusive	contact	contact
Sensor size	small	small	very small	medium	large	medium
Sensor cost	< S200	< \$50	< S5	< \$300	< \$500	< \$300
Template size	< 500	< 1,000	2,000	256	< 100	200
Scalability	high +	medium	low	very high	low	high —

2.3 Technical and processual description of selected method

The basis of the fingerprint recognition is fingerprint - a unique combination of papillary ridges on the fingers. Those ridges on the epidermis are created by dermal papillae on the dermis layer. You can see the structure of human skin in Figure 1.

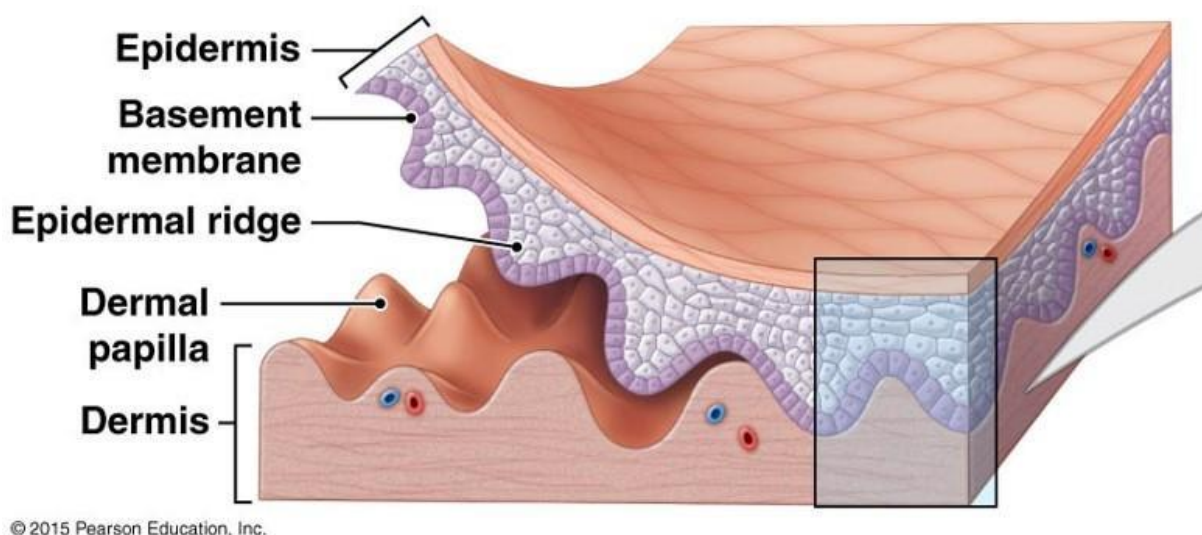


Figure 1: Human skin structure

Human fingerprints are detailed, presumed to be nearly unique, difficult to alter, and durable over the life of an individual, making them suitable as long-term markers of human identity. However, several methods allow to modify or erase fingerprints: burning off with caustic substance, biting off the fingertips, transplanted skin from feet, and surgical removal of fingertip skin. One also can temporarily flatten their fingerprints by abrasion, eg with sandpaper, but the prints will grow back in the same pattern.

A study field, which studies fingerprints, lines, mounts, and shapes of hands is called dermatoglyphics. The scientific process of studying fingerprints for purposes of identification is called dactylography. Exactly dactylography determines all the specific patterns and the recognition methods, defining how and with help of what will different fingerprints be compared, and which attributes will be used to store them.

Criminalistics break the fingerprint structure into 15 different groups: ridge, end ridge, hook, fork, crossing ridge, germination, shift, bifurcation, trifurcation, dot, eye, island, enclosed

ridge, enclosed loop and specialties. However, practically there are two basic methods of fingerprint recognition: pattern based and minutiae based.

Pattern based algorithm compares the basic fingerprint patterns (arch, whorl and loop) between a previously stored template and a candidate fingerprint. A capture device is used to take a graphical image of a fingerprint, typically captured as a TIFF image. The graphical image obtained from the capture device is commonly referred to as a live scan to distinguish it from a template or print stored in a database. Processing software examines the fingerprint image and locates the image centre, which may be off-centre from the fingerprint core. The image is then cropped a fixed distance around this graphical centre. The fingerprint patterns can be seen in Figure 2.



Figure 2: Fingerprint patterns [2]

Fingerprint matching with pattern-based templates involves making a graphical comparison of the two templates and determining a measure of the difference. The greater the difference is, the less likely the prints match.

Minutiae based algorithm, on the other hand, compares ridge ending, bifurcation, and short dots. The ridge ending is the point at which a ridge terminates. Bifurcations are points at which a single ridge splits into two ridges. Short ridges (or dots) are ridges which are significantly shorter than the average ridge length on the fingerprint. After the scanning is complete, special software analyses the fingerprint image and determines if the image actually contains a fingerprint, determines the location of the core, the pattern type, estimates the quality of the ridge lines, and finally extracts minutia. Minutia, from a simple perspective, indicate where a significant change in the fingerprint occurs.

After locating these features in the fingerprint, the minutia extraction software determines a significant direction of the change (using Arrow B as an example, the significant direction starts at the end of the ridge and moves downward. The resultant minutia, in their simplest form are then the collection of all reasonable bifurcations and ridge endings, their location

and their significant direction. Minutia are also assigned a measure of their strength. Visual examples of different fingerprint elements are available in Figure 3.

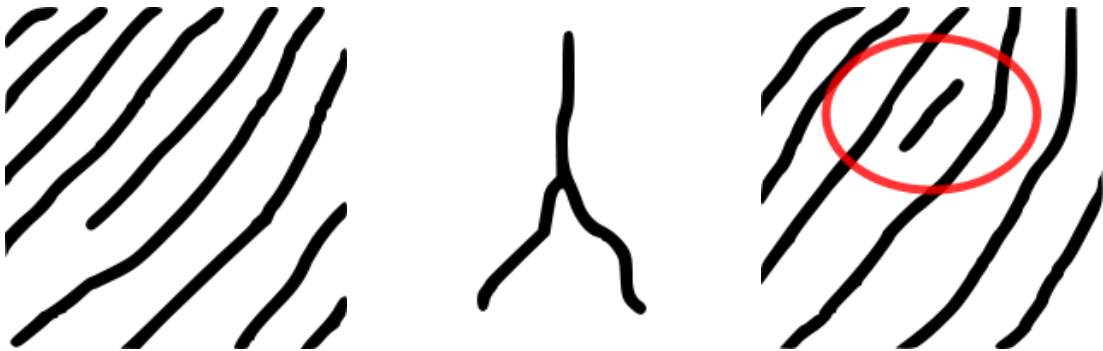


Figure 3: Ridge ending, bifurcation and short ridge (dot) [4]

The singular points of fingerprints, namely, core and delta, are important referential points for the classification of fingerprints. They are used to align fingerprints in case of different scanning angles (Figure 4).

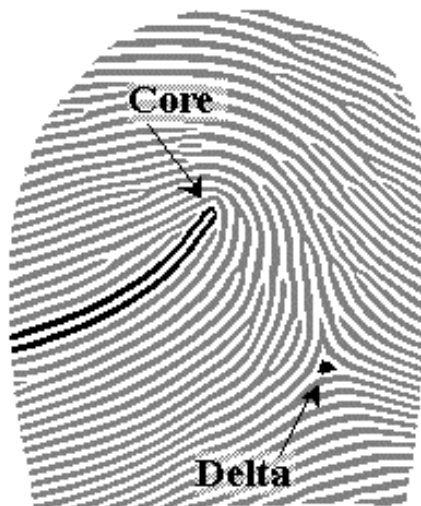


Figure 4: Fingerprint core and delta location [5]

2.4 Algorithms comparison

As it was described in the previous chapter, there are two main algorithms which are used during fingerprint recognition process - pattern-based and minutiae-based. Also different modifications and combination of them are currently developed, but they all are based on those two fundamental ones. In order to be compared, both algorithms should be evaluated

by several criteria: template size and processing speed, sensitivity to the physical changes and security.

Template size and processing speed: On average, minutia-based templates are significantly smaller than pattern-based templates on a byte count basis. The size of a minutia template is directly related to the number of minutia extracted. Identified minutia templates typically average about 350 bytes, or approximately 35 minutiae, but can be as small as 125 bytes. The minutia extraction software is easily able to affect the size of the template by controlling the number of final minutia based on their strength.

Pattern-based templates average about 300-400 bytes when compressed, and about 1024 bytes when uncompressed. Matching and other related functions can only operate on the uncompressed version. However, the size of the template is directly related to the image and cannot easily be controlled without sacrificing detail (and thus usefulness) in the image.

Template size and storage capacity are directly related, with minutia templates requiring about half the storage of pattern template. This impacts storage media costs, network bandwidths, etc., and has a direct effect on the time required to retrieve a template for searching and matching. It also directly relates to the search and match speeds. Although search and match speeds are also dependant on the efficiencies of the algorithms involved, smaller templates will usually result in shorter match time.

Sensitivity to the physical changes: When a minutia based system processes a fingerprint, a scar, fold or other blemish may result in a few minutiae, but these typically represents a small percentage of the total minutia extracted. For example, if 20% of the extracted minutia is disrupted due to physiological changes to the fingerprint since the template was first taken, then there are still 80% of the minutia available for matching. Since a good match can be made with as few as 30% of the minutia, 80% availability provides for a wide safety margin. Minutia templates are therefore very forgiving of physical changes to the fingerprint without having to resort to re-extracting a new template from a new image of the finger.

On the other hand, pattern-based templates are more sensitive to physical changes in the fingerprint because the match is done using a cropped fingerprint image. Physical changes can obscure critical elements of the image and significantly increase differences between two images of the same finger, thus reducing the likelihood of obtaining an accurate match. In a

pattern-based system, new scars or other blemishes typically require a new image of the fingerprint be obtained, converted to a template and stored in the system.

Security: Security is a major consideration when discussing template types. A possible technique to circumvent fingerprint biometric security would be to obtain an actual template and replay it to the authentication system.

When a minutia-based template is extracted from a fingerprint, subtle variations in the orientation and centring of the finger on the capture device have subtle effects on the minutia generated. This means that the same finger placed on a capture device multiple times will produce slightly different minutia templates each time. This has no effect whatsoever on the accuracy of the matching algorithms (as previously discussed, minor variations in the minutia do not affect the match outcome). Consequently, the same finger presented multiple times will match, but not perfectly in the sense that the extracted templates will never be absolutely identical. This directly leads to a method for detecting the presentation of a stolen template: if the match is exact, the template must be an identical match, minutia for minutia, with the template in the database. The template must therefore be a duplicate of the one in the database and could not have come from a live scan.

With a pattern-based template, obtaining the template, since it is a cropped graphical image of the fingerprint, gives you the actual fingerprint. Adding logic in the authentication system to detect that the exact same fingerprint image is being presented increases the False Reject Rate, that is, the number of valid users being rejected. Further, the stolen print from the template can be subtly altered so as to prevent a duplication detection, but still result in a positive match. With a minutia template, the fingerprint cannot be reconstructed, and thus the fingerprint itself cannot be subtly altered and then replayed to the authentication system.

3. System proposal

This chapter reviews, how the chosen biometrical method may be applied, combined with the existing RFID solution and implemented. It contains technical description of the components, including requirements for software and hardware, component selection, architecture analysis and proposal.

3.1 System architecture

Building a properly functioning system requires strong system architecture definition. Not only it gives the clear view on what should be done, but also defines which elements does the system contain, how do they work, communicate, which inputs and outputs do they have. Having system architecture overview also reduces costs on the its implementation and the time losses.

There are four possible architectures for this system. The main difference between them is fingerprint storage: in the first and the second case, it is database, in the third one - RFID card, in the fourth one - the reader itself. Also, the first and the fourth concept generally (in some cases) do not require RFID card at all, as fingerprint recognition has sufficient security level. The architectures are analysed according to this aspect, since in everything else the structure stays the same.

- *Database 1*: fingerprint scan is kept in the central (or split to several local ones) database. The solution does not necessarily require RFID card. The evaluation process in this case has the following sequence:
 - User attaches his finger to the scanner
 - The scan is encoded and sent to the server
 - Server seeks the fingerprint and sends the response to the scanner
 - The scanner either grants the access or declines
 - The card might be used as an additional proving attribute, which is, however, not necessary at all
- *Database 2*: fingerprint scan is kept in the central (or split to several local ones) database. It is also additionally encoded by the unique key, which is stored on the RFID cards. The evaluation process in this case has the following sequence:
 - User attaches both his finger to the scanner and RFID card to the reader
 - The data from the card are decoded

- Fingerprint scan data and RFID key are encoded by the reader and sent to the server
- Server receives the data and decodes them
- If user is not found in the database or is restricted from this area, negative response is sent immediately
- Elsewise the database-stored fingerprint is decrypted using the RFID card key, and compared with the received one.
- Based on the comparison, server sends the response to the reader
- *Reader:* fingerprint scan is kept directly on the reader. The solution does not necessarily require RFID card. The evaluation process in this case has the following sequence:
 - User attaches his finger to the scanner
 - The scan is encoded and compared with the records in reader's memory
 - The scanner either grants the access or declines
 - The reader might be used as an additional proving attribute, which is, same as in the previous solution, not necessary at all
- *RFID card:* in this case, fingerprint data are stored in the card (ISIC, CTU card, etc.). The evaluation process in this case has the following sequence:
 - User attaches both his finger to the scanner and RFID card to the reader
 - The data from the card are decoded
 - Fingerprint data are extracted and compared with the real fingerprint
 - Simultaneously, other data part, such as user ID, are sent to the server
 - The server checks user's presence in the database and his privileges, then sends the response to the reader
 - Reader gets the response. In case it is positive, and the fingerprint comparison also gives positive match, the user is granted an access

The second, third and fourth architecture appear to be the most secure due to several reasons.

The second architecture is secure due to the following reasons:

- Despite communication links might be cracked by the hackers, this will give no useful information to them
- Even having the access to the database, hackers still won't receive any vulnerable information, as every fingerprint is encrypted by the unique key on every RFID card
- This way, getting fingerprint database requires having ALL the cards keys, which is practically impossible to get

- To get the access to the area, hackers need to complete several actions on the different architecture levels

The fourth architecture (also called “Match-on-card”) is the more secure than the others because of the several reasons:

- Fingerprints are not directly sent to the server, as it is in offered in the first solution. Despite they are still encoded, such transfer may bring additional risks to the system.
- Fingerprints also are not stored directly in the database. Even being encoded, it is still a threat since there is a chance of getting public or private database key.
- Fingerprints are not stored locally, which excludes the chance of getting the whole database with the key by simply retrieving data from a single reader.
- In case the changes need to be done (ex. some users are to be restricted from the access) there is no need to repeat the process over all the readers. The access restrictions are done on the database level.
- In case server is attacked, hackers will not receive any sensitive data.
- Easier to implement and to test, as less network communication is required

Comparing to the second option, fourth one is still more secure what concerns connectivity. It also has almost all the processes done locally, server plays simple background role. However, exactly at this point the problem may occur: not all the types of processors are able to handle such evaluation task. The decision on which solution should be used depends on the estimation results of network, reader and server processor’s capabilities.

3.2 RFID

3.2.1 Technology description

Radio Frequency Identification (RFID) is an identification technology, initiated by Wal-Mart, which is nowadays used for various industries, including logistics, pharmacy, food, aerospace, human identification, etc. It uses short-distanced communication using electromagnetic fields on following frequencies: 125 kHz, 134 kHz (low), 13.56 MHz (high, ISO 18000-3) and 860-900 MHz (ultrahigh, ISO 180000-6c).

There are two main types of RFID tags: active and passive. Active RFID tag uses energy source from the battery, and periodically transmits the signal. Passive RFID tag is activated by the RFID reader with its radio energy. Third type - battery-assisted passive tag - has a small battery, but transmits the signal only after receiving the awakening signal from an

active reader. Also, depending on the possibility of rewriting tag information, it might be either read only or read/write.

The tag chip construction has four main blocks:

- *RF front end*: responsible for power recovery, demodulation of the incoming RF signal, and the backscatter transmission of return data.
- *Analog part*: generates the system clock, current/source reference, and power-on-reset (POR) signal, etc.
- *Base-band section* and *EEPROM memory*: handle power management, data recovery, operating protocols, and user-available data storage.

RFID reader might be either active (ARPT or ARAT) or passive (PRAT) depending of which cards is it designed to communicate with. Active reader this way may be designed to communicate either with passive or active tags. In the first case, the reader transmits interrogator signals and receives authentication replies from passive tags, In the second one, interrogation signals are much weaker, and are used only to awake active or battery-assisted passive tags. Passive reader only receives the signal from active tags. [6]

3.2.2 RFID system at CTU

CTU uses active readers, which communicate with local server, and passive RFID tags, which are implanted in identification cards. CTU's tags are produced by NXP Semiconductors (Eindhoven, Netherlands), and belong to Mifare DESfire family, exactly EV1. Card readers depend on area of usage. The most widespread one, which is used to control the access to the rooms and locations, is iClass RW100 reader from HID Global (Austin, TX, USA).

HID iClass RW100 communicates on 13.56 MHz frequency. It supports DESFire EV1, up to 32k bit user credentials data, including IT secure authentication, biometric template storage, digital cash, time and attendance, equipment/material use or checkout and transit passes. Output comes in Wiegand protocol format. For communication with PC or microcontroller the reader used RS-232 serial port. It also corresponds to ISO 15693 standard. [7]

Mifare DESfire, first introduced in 2002, is a family of RFID tags, which use DES, 2K3DES, 3K3DES and AES hardware cryptographic engine for securing transmission data [6]. The first generation (deprecated in 2010) had triple-des (2K3DES) encryption and 4k bit storage. EV1, which is actually used by CTU, uses AES encryption, has 2/4/8k bit memory (CTU uses 4k

bit modification) featuring also additional security features, such as CMAC. It is certified with Common Criteria EAL4+. It features an on-chip backup management system and mutual three pass authentication, allowing it to hold up to 28 different applications using MAD3 and 32 files per application. Mifare DESFire EV1/2 are fully compliant with ISO/IEC 14443A. [8,9]

MAD (Mifare Active Directory) standard proposes the introduction of common data structure for card application directory entries. Version 1 is limited to 16 sectors, version 2 specifies the usage of the MIFARE with a memory >1kb, version 3 specifies application usage. Standard has 3 types: monoapplication card without directory entries and multiapplication card without/with directory entries. CTU uses the last one, as the card is used for several purposes, including access to the rooms and payments for different services (food at the canteen, copy services, administrative payments, etc.). [10]

The MIFARE DESFire card IC features a flexible file system which organizes user data in applications which hold files. Applications are identified with a 3 byte application identifier (AID). AIDs have to be unique per card and are defined at application creation time. MIFARE DESFire IC maintains application list automatically. Mifare DESFire EV1 has native support of MAD3.

For every application directory there are: application identifier (16 bit), CRC-byte (8 bit), info-byte (8 bit, card distributor information), general purpose byte (8 bit, contains further standard details), read-key A (5 bit, open) and write-key B, which is ought to be kept in secret, as it gives the access to modify the data. [11]

Generally the card specifications (exactly Mifare DESFire EV1) must be supported by new system's reader in order to be compatible with the existing cards on CTU. The issue, which needs to be thought through is how to store fingerprint data into the card.

3.2.3 Instructions language

In chip cards domain, a specific communication unit between the card and the reader is used. It is called APDU (application protocol data unit). The structure of APDU is defined by ISO/IEC 7816-4. APDU may be either a command or a response.

During the communication, command APDUs are sent by the reader. Above up to 64kb of data, it contains a mandatory 4-byte header of structure [CLA, INS, P1, P2]. Response contains up to 256 bytes of data and compulsory 2 bytes of status code, indicating (in

hexadecimal form) request status (ex. if it was completed, or there has occurred some error). More wide description of APDU can be found in section 3.6.

3.2.4 Reader provided

The reader provided is STid ARC-A, with a fingerprint module Architect SE6. The reader supports most of the Mifare standards, including DESfire EV1. The fingerprint module may be unattached from the ARC-A. Also, a separate reader with USB cable comes with the SDK. This version of the reader, however, does not support "Match-on-card", so the fourth solution architecture is not applicable. That means that the architecture is going to be based on a local fingerprint pattern hash storage (i.e. reader's internal memory).

3.3 System analysis

In this part system elements, functionality, goals, risks and issues are described in details

3.3.1 System elements and their functions

Card - CVUT or ISIC identification card with RF microchip, which holds basic information about its owner: name, faculty, membership, student's account balance. Within the new system it will also carry encoded fingerprint.

User - student, teacher or worker at CVUT, who will be the main user of the system. In most cases owns the card. In some User Cases may be replaced by Deceiver - a person which somehow tries to deceive the system.

Reader - device, which reads the data from the Card. Also stores pair <Card UID, Fingerprint hash> on the internal memory. UID is sent to the Server in order to check, whether the UID is valid or not.

Scanner - fingerprint scanning device, which will scan User's fingerprint and send it to the Processor.

Processor - the CPU, which will run the APDU commands.

Code - set of APDU instructions, most often implemented in more widespread programming languages, which define the authentication procedure sequence

Receiver - device, which will require the result sent by the Applet and use it for generating the output, which will be sent to User. According to the situation it might be computer, door lock or something else.

Host - computer, communicated to the Reader, which sends requests and controls its actions.

Table 3: System components

Element	Content	Function	Communicability
Card	RFID card with chip, user data	discrete, deterministic	Yes
User	Biometrical data	discrete, deterministic	Yes
Reader	RFID card reader, wire	continuous, deterministic	Yes
Scanner	Fingerprint reader, wire	continuous, deterministic	Yes
Processor	CPU unit, drivers	discrete, deterministic	Yes
Data	NAND memory	continuous, deterministic	Yes
Code	APDU instructions	discrete, deterministic	Yes
Host	PC	continuous, deterministic	Yes
Receiver	?	?	?

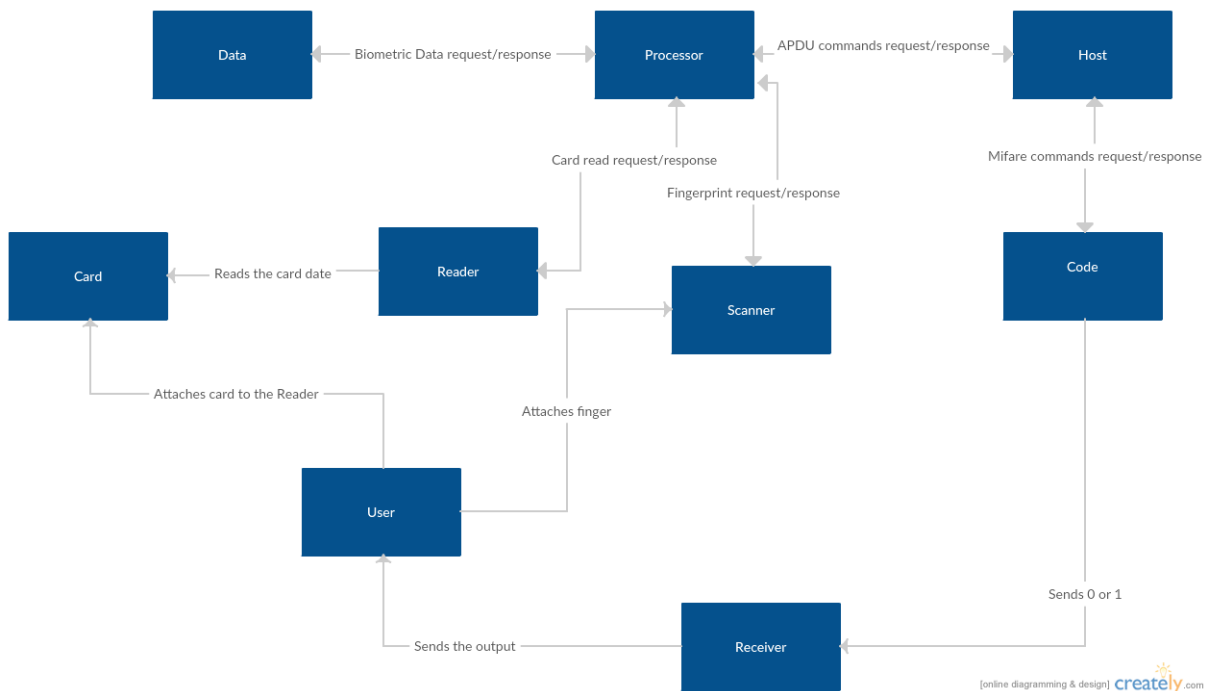


Figure 5: System component diagram

3.3.2 System's competence

System's main functionality is to authenticate and authorize CTU's students, teachers and staff in order to enable them different actions. Those actions might be different, however, the main purpose is locality access restriction. Potentially the system might be used for student authorisation during the exam, or, for example, while assigning the subject list on the study department.

3.3.3 Activity diagram

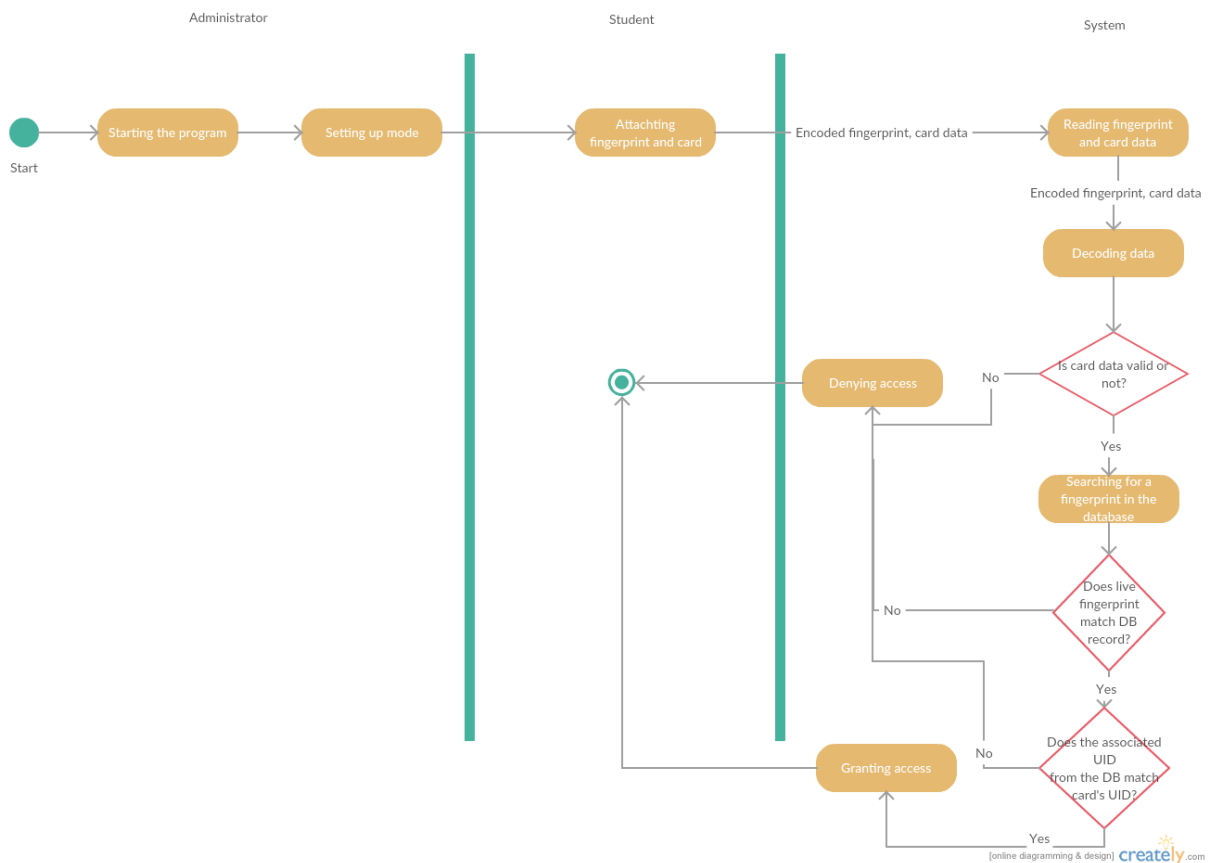


Figure 6: Main activity (authentication) diagram

3.3.4 System requirements

Before the architecture of the system is defined, it should be clear, what is required of the system. The potential unfulfillment of those requirements might lead to wrong/unpredictable functionality, errors or even dysfunction of the system. In case of that simple system, the only requirements are:

- Power supply: +7 Vdc up to +28Vdc;
- Communication: possibility to connect the reader to the server (via RS-485)
- Dust/water impact: max. IP65

3.4 Software Development Kit (SDK)

3.4.1 General Description

The software development kit (SDK) used for STid readers is DEVKIT GLOBAL by STid. It is possible to develop software for Mifare Classic/Plus/DESFireEv1/UltraLightC, CPS and biometrics modules on it. The development kit enables data confidentiality between the reader and the controller, with the secure communication protocol SSCP (STid Secure Common Protocol). This protocol offers data encryption (AES) and mutual authentication (HMACSHA-1) prior to any communication. Example software (SESPro) is a great example [7] The mainframe and the interface of the software can be seen in Figure 7.

User can select the connection interface (RS232/485, TCP/UDP, USB), and also import or export connection settings to a file. In “Crypto level” tab it is possible to change the connection security settings, such as security mode (plain, signature, enciphered, both or SSCPv2), authentication settings and keys. “Reader” tab represents general reader settings for LED mode, keyboard and tamper switch. “ARC” tab has more LED colour setting



Figure 7: SESPPro interface

Mifare DESFire EV1 settings are shown in Figure 8. Security settings define authentication type, key location, encryption method and master key settings. Application settings contain

settings for applications, their name, addresses, ISO files. File and Data tabs have all necessary read/write/modify/encrypt/etc. functionality for both files and pieces of data.

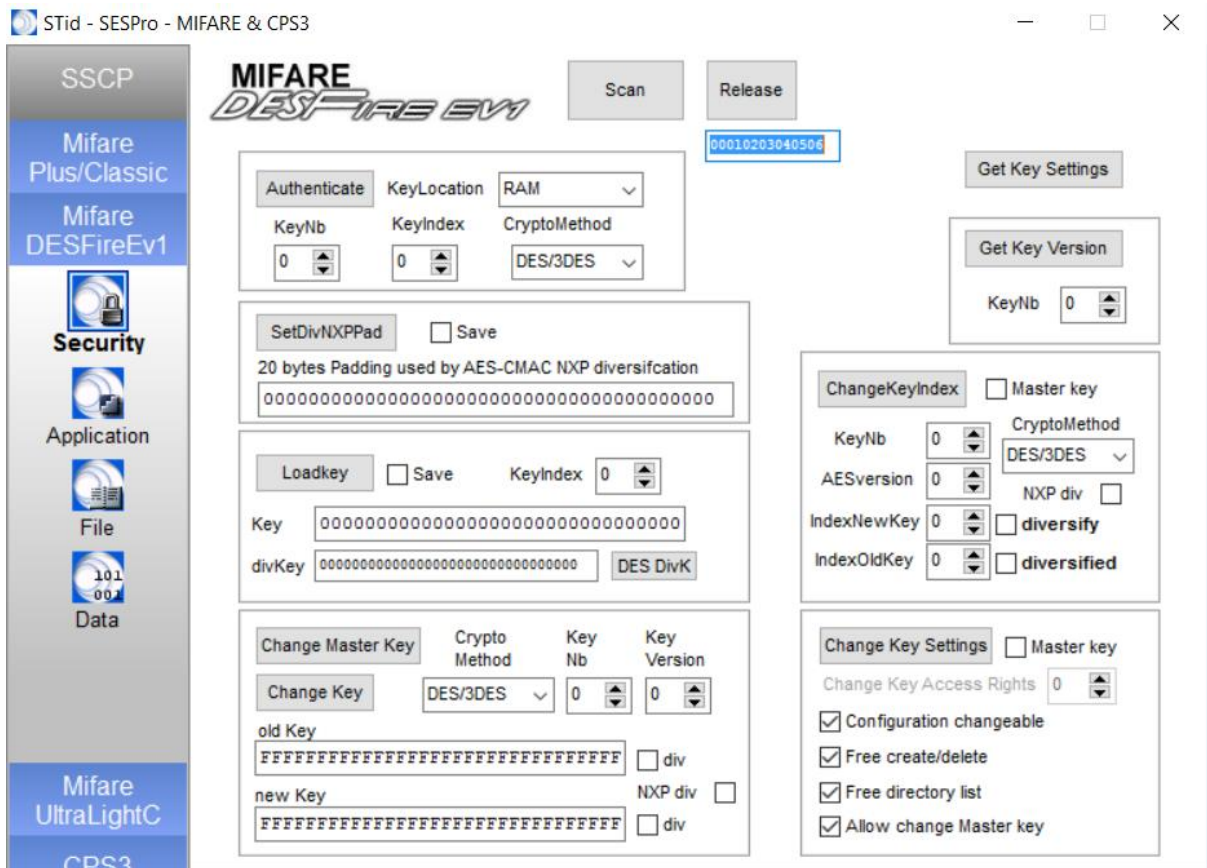


Figure 8: SESPro Mifare DESFire EV1 settings

The general SDK description can be found at [12].

3.4.2 NPX APDU

Before describing APDU itself, a quick view on card's file structure should be given.

DESFire EV1 has a tree-organized data structure. Highest hierarchy level units on the card are applications, which have different targets: identification, payments, authentication, etc. The applications, on the other hand, may contain several files, which are used for applications' purposes. The files are defined by one or more data sets, which have exact memory addresses, and represent the lowest instance levels of the card structure. For every layer, there is a different set of APDUs predefined, depending on the version used. The hierarchy itself is shown in Figure 9. APDU response has more simple structure: response data are finished with 2 bytes, which represent response status hexadecimal.

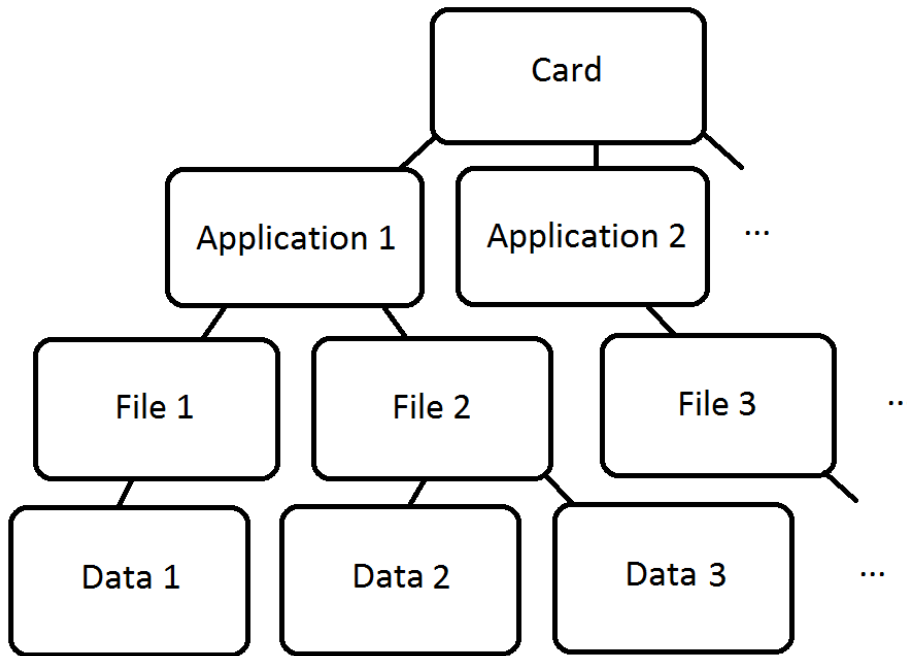


Figure 9: DESFire EV1 card structure

Full command APDU structure is shown in Table 4 [11].

Table 4: Command APDU structure

Field	Description	Section
CLA	Class byte 0x00	Header
INS	Instruction byte 0xA4:Select 0xB2:Read/Write	Header
P1, P2	Parameter 1 and 2 bytes (INS).	Header
Lc	Number of bytes sent	Trailer
Data	Data byte	Trailer
Le	Number of bytes expected in the response.	Trailer

APDU instructions themselves, however, are extremely unintuitive. They are formed by hexadecimal set of numbers, so basically almost every developer could have problems in developing direct APDU code. Example of an APDU command is

A0 FA 00 00 00 [13]

which is “sleep” command.

That is why modern applications, which work with card readers, use already implemented libraries with sets of commands written in more widespread and intuitive programming languages.

3.4.3 MifareGlobal library

SDK provided by STid contains not only example software, such as SESPro, but also .dll library with all Mifare Global commands, and several wrappers in different programming languages, including C++, Python and C#. Also example software, functionally similar to SESPro, written in Visual Basic .NET and Delfi are included. Those give the wide view on the possibilities of the client-side software, which could be developed using this library.

The code for this project is developed in C#. Nevertheless, assuming that other developers may have different experiences, it is possible that for real-life implementation of the system another language might be used.

3.4.4 Documentation

The documentation file describes all implemented functions from MifareGlobal dll library. It explains which functions and when should be called, required inputs and outputs, same as their formats. It uses Pascal/Delphi style code, so despite the names of the functions stay the same, some transformation of the variables needs to be done in order to make it work in other programming languages.

All functions are separated into several sections, which are:

- General functions: basic functions, which define connection settings with the reader and session settings
- Reader functions: general reader functions, defined independently on the card platform used (LED and buzzer control, crypto, etc.)
- Mifare DESFire functions: functions exactly for DESFire and DESFire EV1 platform (card, security, applications, file, data)

- Mifare PLUS functions (security levels 0-3)
- Mifare UltraLight and UltraLight C functions
- CPS3 functions
- Biometric functions

Basically every function description contains function name, its inputs description and their types, general function description including some practical advices. Also error codes for all functions are provided in appendix. However, except from practical examples which are mentioned before, there are no other testing code pieces. The whole documentation file can be found in Attachment 2.

3.5 Program

In this section the code which uses RFID and fingerprint reader is described, including development issues, correspondence with the system's architecture, functions used, and finally testing results.

3.5.1 Development issues and limitations

First of all, what needs to be said is that the final form of the code drastically differs from how it was originally imagined. When the project started, it was thought that the code itself will be written directly for the reader, not for a host PC. However, after the reader and SDK have arrived, it was discovered that there is no direct possibility to create a code for reader's processor, so the basic idea of implementing Match-on-card biometrics has been refused. The absence of the possibility to use basically preferred solution has lead to re-thinking the whole system architecture, and this way significantly increased the role of the host PC and reader's storage. Also, before the understanding of impossibility of writing the code directly for the reader, it was unclear how to implement the command sequence (exactly which language should be used).

Some problems were also discovered during STid's software exploration, including sudden crashes and problems with connection and card reading or authentication failures. For example, the program could somehow simply "forget" already authorized card, so the command started to return error codes, and the card should have been re-authorized in order to continue the job. Example software has behaved even worse. Probably those problems were caused not only by the programs themselves, but also by the connection issues. STid somehow didn't implement direct USB connection support (about which manufacturer however warns in SESPro), so the reader used for the testing, despite being connected via USB cable, hasn't been recognized by the program first. The solution which they advised

(and which was used for this project) was to use virtual serial COM port driver, which recognized the reader connected through USB port as a COM device.

3.5.2 Language selection and pre-requests

As already mentioned before, the language chosen for applet implementation is C# with .NET framework. Nevertheless, it is possible basically to use any language - it only needs writing a wrapper for the original functions from the DLL file.

The C# wrapper is located in "Lib" folder. It is a single .cs (C# file format) file, which contains the list of all Mifare Global functions using C# code format and types. In order to understand the transferred function, the user has to take a look at the types used in manual and Table 4, where data types from manual-used format (Pascal/Delphi) is matched to the corresponding data type in C#.

Table 5: Data formats [Att. 2]

Type in manual (Psc/Delfi)	Explanation	Dimension	Corresponding type in C#
Byte	8 bits	[0..255]	byte
Word	Unsigned 16 bits	[0..65535]	UInt16
Integer	Signed 32 bits	[-2147483648..2147483647]	int
Cardinal	Unsigned 32 bits	[0..4294967295]	UInt32
PChar	signed char[]	-	string
TDataByte	array of byte	[0..1023]	byte[]
TDESFireKey	array of byte	[0..15]	byte[]
TMIFAREKey	array of byte	[0..5]	byte[]
TAESKey	array of byte	[0..15]	byte[]
TSSCPmode	one byte	(0,1,2,3)	byte
TSSCPcomtype	one byte	(0,1)	byte

The output variables, which should be predefined in code before running the method, have parameter **ref** before them in function definition. The ones which do not require predefinition use **[Out]** modifier. Predefinition in this case means not reserving variable name (this should be done in any case though), but assigning any value to the variable. So, for example, to use function

```
static void FillArray(out int[] arr)
{
    arr = new int[5] { 1, 2, 3, 4, 5 };
}
```

it is enough to write

```
int[] theArray;
FillArray(out theArray);
```

while to run this function

```
static void FillArray(ref int[] arr)
{
    if (arr == null)
    {
        arr = new int[10];
    }
    arr[0] = 1111;
    arr[4] = 5555;
}
```

a value should be assigned

```
int[] theArray = { 1, 2, 3, 4, 5 };
FillArray(ref theArray);
```

This plays an important role in code development, as it becomes a crucial need to be careful and exact with the variables definition, always making sure that it is properly initialized and - for the arrays - it fulfils minimum size needed for the function, because in the other case it will return an error code.

3.5.3 Code functionality

According to the system concept, the applet's code is executed directly on the host, i.e. controlling PC. Its realisation should reduce human's interaction with the host to minimum, so main runtime process must be somehow automated. Connection functionality between CIPS servers and host PC will not be implemented, due to open architecture option (the user database may be either on a remote server or on a local). This way, the code itself is ought to do the following:

1. Establish connection with the card reader and authenticate it
2. Create/erase local fingerprint database
3. Add/remove user's fingerprint on/from the local database
4. Authenticate the user using his fingerprint and card data
5. Close connection/exit

The first method needs to be launched manually by the user. However, calling the corresponding function from the main method will reduce the need of human interaction with the program to a single launch of the application. The second method nevertheless must be called by the user directly, as MifareGlobal library does not contain any method to check whether the database already exists or not. The other two actions might be implemented as automated. In order to make functionality more exact, Activity Diagram must be used. According to it, the action sequence is to be the following:

1. Administrator starts the script on the PC after the reader is connected
2. The script initializes connection with the reader and authorizes it.
3. Continuous automated process starts
 - 3.1. Reader waits until user places the card
 - 3.2. Card's UID is sent to CIPS server in order to check, whether the card is registered and whether the user has an access to the room
 - 3.2.1. In case of the positive response, the script reads the fingerprint and checks if it matches the fingerprint stored on the database which is associated with the following UID

However, this is not the most detailed description. Also, in order to make the functionality more autonomous, the users, who enrol the system for the first time, must be registered automatically, so that administrator does not need to add every user manually. But during the investigation it was discovered that unfortunately Mifare Global functions do not dispose with the functionality which is enough to automatize the registration process. The core function

missing is a function, which would have returned either fingerprints associated with a card UID (in this case, the output value should have been a set of byte arrays representing fingerprint data) or at least a sign of presence of any records in database associated with this UID (return value would be Boolean true/false). As there is only a function which identifies live fingerprint match within the database, there is no way to check whether the person holding the card is its real owner - the identification function would simply return that there's no records associated with the live fingerprint, so the algorithm would decide to add the record in the database, assuming that it's the first time when the user is logging in. Taking into consideration this fact, the full action sequence within the mainframe looks like this:

1. Administrator starts the script on the PC after the reader is connected
2. The script initializes connection with the reader and authorizes it.
3. Continuous automated process starts
 - 3.1. Reader waits until user places the card
 - 3.2. Card's UID is sent to CIPS server in order to check, whether the card is registered and whether the user has an access to the room
 - 3.2.1. If the response is positive, the script scans the fingerprint and checks if it matches any of the fingerprints stored on the database
 - 3.2.1.1. If the fingerprint matches the local one, the algorithm checks whether the returned UID matches the card's UID
 - 3.2.1.1.1. If yes, the reader buzzes and blinks green, and the door is opened
 - 3.2.1.1.2. Else the reader buzzes and blinks red, the door remains closed
 - 3.2.1.2. Else the reader buzzes and blinks red, the door remains closed
 - 3.2.2. Else the reader buzzes and blinks red, the door remains closed

Also separate functions should be defined for:

1. Adding record to the fingerprint database
2. Erasing record from the fingerprint database
3. Creating fingerprint database
4. Erasing fingerprint database
5. Opening the door (not required to be implemented)
6. Checking UID validity (not required to be implemented)
7. Closing the connection

Basically, this is everything what needs to be implemented. Some other functions, ex. datatype converters, which might appear to be required, will be added dynamically. It is also obvious that the script should be using import from the original wrapper by STid.

3.5.4 Code implementation

It is logical to start with the “main” process. But even before that, the wrapper needs to be used, as it is situated in another class. This is easily done by one simple row

```
using SSCPLibMIFARE_Wrapper;
```

Note that the file SSCPLibMIFARE_WRAP.cs needs to be included in the project in order to make the system work.

The main method (i.e. initialisation) consists of the following steps:

1. Initialisation of the library
2. Connection with the reader
3. Launching the reading block

The code for the block is the following

```
public void on(int mode){ // represents the mode in which the reader is to be
used: 1 stands for adding record, 2 stands for removing records, 3 stands for
normal use (authentication)
    SSCPLibMIFARE.SSCP_Initialize();//Mifare DLL library initialisation
    SSCPLibMIFARE.SSCP_SetMode(1);//Setting up the connection mode to
authenticated
    check(mode); //launch the reading block in the selected mode
}
```

The reading block functionality is described in the previous chapter. The code for the frame is

```
public bool check(int mode){
    byte RATS; //indication byte, which tells whether the ATS information
has been sent
    UInt16 ATQA = 0;
    byte nbCard, SAK, UIDlen = 0x00; //nbCard - indicates the presence of
the card; ATQA - chip's Answer To Request, SAK - chip's Select Acknowledge,
UIDlen - chip's UID length in bytes;
    byte[] UID, ATS = null; //UID - chip's UID; ATS - reader's response
to RATS
```

```

        while (nbCard != 0x01){ // while no cheap has been read
            SSCPLibMIFARE.SSCPr_Scan_A_Raw(RATS, ref nbCard, ref ATQA, ref
SAK, ref byte UIDLen, [Out] byte[] UID, [Out] byte[] ATS); // scan for a
cheap; this function is a safe-for-loop alternative for
SSCPr_Scan_14443_A() function
        }
        if(!checkUID(UID)){ //call for a server in order to indicate whether
the chip with this UID has an access to the place (not implemented)
            SSCPLibMIFARE.SSCPr_Outputs(2,100,100); // red blink (2) and
buzzing to indicate authentication failure to the user
        }
        else{
            if(mode == 1){ //if the mode is set to registration
                addBioRec(UIDm UIDlen); //add biometric record to
database
            }
            else if(mode == 2){ //if the mode is set to removing
                removeBioRec(UID, UIDlen); //remove the record
            }
            else{
                byte NBAattempt = 0x01; //one attempt given for
fingerprint identification
                byte Match, UID2len = 0x00; // Match - identifies
whether the attempt has been successfull, UID2len - length of UID
associated with the fingerprint
                UInt16 TimeOut = 1; //timeout for the search in seconds
                UInt32 RFU = 0; // 4 bytes set to 0
                byte[] UID2; //result UID of the card associated with
fingerprint (if exists in database)
                SSCPLibMIFARE.SSCPb_Identify(NBAattempt, TimeOut, ref
Match, ref RFU, ref UID2len, [Out] UID2); //searches for the live
fingerprint match in database
                if(Match == 0x00){ //if no match has been found
                    SSCPLibMIFARE.SSCPr_Outputs(2,100,100); // red
blink (2) and buzzing to indicate authentication failure to the user
                }
                else{
                    if(UID2.SequenceEqual(UID)){ // if UID of the
matching fingerprint is the same as card's UID

```



```

        byte[] curr_template = new byte[temps[index]]; //creating new array
        for a template
        int length = Convert.ToInt32(temps[index]); //getting template length
        Array.Copy(temps, index, curr_template, 0, length); //retrieving template
        value to the empty array
        SSCPLibMIFARE.SSCPb_AddRecord(1, curr_template, UID_length, UID);
        //stores current template in the database
        index = length; //saving current position
    }
}

public void removeBioRec(byte[] UID, byte UIDlen){
    SSCPLibMIFARE.SSCPb_RemoveRecord(0x00, UIDlen, UID); //removes records for
the UID
}

```

The only functions left are: creating and erasing database, stopping connection and two unimplemented functions - one for door opening and one for UID check on the server.

```

public void createBioDB(UInt16 size){
    SSCPLibMIFARE.SSCPb_CreateDB(size, 0x03); //create database for n user with
maximum of 3 fingerprints per user
}

public void eraseBioDB(){
    SSCPLibMIFARE.SSCPb_EraseDB(); //erase the database records (but not the
database generally)
}

public bool checkUID(byte[] UID); //method used to check whether the card's holder
has an authority to perform action (ex. open the door)

public void doorOpen(); //method opening the door (may be replaced by any other
method which is the result of positive authentication)

public void off(){
    SSCP_Disconnect();
    SSCPLibMIFARE.SSCP_Terminate();
}

```


Basically, those are all functions necessarily needed in order to maintain a combined RFID+fingerprint system.

3.5.5 Notes for the code

As already mentioned before, the following code is only an approximated basic example of how the code for the following reader might be written. It is developed taking into consideration some of the limitations of Mifare Global SDK library.

Some complications were also called by the documentation, i.e. functions description were not always precise and understandable, some of the variable types were missing or were mixed up.

Of course, some of the points in the code bring doubts about how optimal it is. In this part of the project those points will be explained.

- 1) Initialisation procedure: it might probably happen, that after the initialisation it is better not to start check() procedure automatically. As it has a loop inside it, and the function itself is recursive, it can appear that after the initialisation it will be impossible to run other functions, such as create/erase database and so on. Which means, that in this case function check() should be launched manually by the user. The initialisation and evaluation procedure launch process now basically takes 1 command:

```
on(3)
```

But in case we change its code to

```
public void on(){ // represents the mode in which the reader is to be used: 1
stands for adding record, 2 stands for removing records, 3 stands for normal use
(authentication)
    SSCPLibMIFARE.SSCP_Initialize();//Mifare DLL library initialisation
    SSCPLibMIFARE.SSCP_SetMode(1);//Setting up the connection mode to
    authenticated
}
```

the following should be done to launch evaluation:

```
on()
check(3)
```

2) This however does not solve the possible issue of impossibility to launch termination operation while recursive reading runs. In this case, the possible solution might be to declare a global variable at the start (outside the functions)

```
bool flag = true;
```

Then, the recursive call of check() inside the function itself should be wrapped in the “if” condition

```
if(flag)
{
    check(mode);
}
```

And when the user wants to stop evaluation process, he needs to change value of flag variable in console.

3) Usage of the “while” loop in order to continuously execute the “read card” command might be not optimal and make the system less power-efficient. Documentation, however, mentions that the function used in the code (SSCPr_Scan_A_Raw) is suitable for a loop, contrary to SSCPr_Scan_14443_A. Nevertheless, it is also mentioned after, that the reader may be put into the autonomous mode, when the reader only returns periodically frames, which may (but not must) contain the UID of the card. In this mode, the reader only accepts one command type from the host, which brings it back from the autonomy. The settings of the frame are also described in the documentation, separately for every card type.

The problem however is that there is no exact description, how those frames are received. Neither there is any example of usage this mode. The example code which comes with the reader describes only the initialisation of the mode, not the way to work with it. In case STid somehow updates the SDK package and brings some clearance in usage of the autonomous mode, it might be possible to change the function in a proper way. So the variant used in the current implementation is possibly not the best due to energy-efficiency, but at least it should work.

4) Possible loop error while enrolling new fingerprint: despite the number of the fingerprints is set up while launching SSCPb_Enroll, it is not clear from the output, whether the idea of storing all three templates in one array works fine. In case it does not work well, all three fingerprints should be enrolled separately, and the addBioRec() method code will look the following way:

```

public void addBioRec(byte[] UID, byte UIDlen)
{
    byte NBTemplates = 0x00; //return value representing the number of the
    following template
    UInt16 temp_sizes = 0; // return value representing total templates size
    byte[3][] temps = new byte[3][1024]; // templates storage
    UInt32 inde
    byte RFU, RFU2 = 0x00; //default values set to zero
    UInt16 timeout = 1; //timeout setup for one second
    byte nbOffFinger = 0x03;//total number of fingers to be enrolled
    byte minutiae = 0x01;//this value indicates whether to exclude calculated
    minutiae; it brings faster search but requires more space.
    for( int i = 0; i < 3; i++)
    {
        SSCPLibMIFARE.SSCPb_Enroll(RFU,timeout,RFU2,nbOffFinger,minutiae,UIDle
        n, [In] UID, ref NBTemplates, ref temp_sizes, [Out]
        temps[i]);//extract the template
        SSCPLibMIFARE.SSCPb_AddRecord(1,temps[i],UID_length,UID); //stores
        template in the database
    }
}
}

```

Nevertheless, from the documentation text is is also still not clear, whether SSCPb_Enroll method automatically adds the fingerprint template to the database or not. It is assumed that it does not. However, in the opposite case usage of SSCPb_AddRecord method will be unnecessary, which means that the code of addBioRec() method shall reduce to the following form:

```

public void addBioRec(byte[] UID, byte UIDlen){
    byte NBTemplates = 0x00; //return value representing the number of the
    following template
    UInt16 temp_sizes = 0; // return value representing total templates size
    byte[] temps; // templates storage
    byte RFU, RFU2 = 0x00; //default values set to zero
    UInt16 timeout = 1; //timeout setup for one second
    byte nbOffFinger = 0x03;//total number of fingers to be enrolled
    byte minutiae = 0x01;//this value indicates whether to exclude calculated
    minutae; it brings faster search but requires more ROM
}

```

```

SSCPLibMIFARE.SSCPb_Enroll(RFU,timeout,RFU2,nbOffFinger,minutiae,UIDlen,
[In] UID, ref NBTemplates, ref temp_sizes, [Out] temps);//extract the
template
}

```

5) Checking error codes: this code has been developed as an example only, so its main target is to give a sight on a functionality required for basic usage of combined biometrics and RFID cards. It is minimized in some ways, as it has to be viewable for other programmers, and that is why it does not include handling response status codes, which would otherwise be necessary for avoiding unexpected crashes in case of connection problems. In code this would illustrate as follows: every SSCP function returns a status, so any launch would look like

```

string msg = "";
UInt16 status = SSCPLibMIFARE.SSCPx_function(arg1, ...);
if ((byte)status == SSCPLibMIFARE.SSCP_OK)
    {
        msg = "Success"
    }
else SSCPLibMIFARE.SSCP_GetErrorMsg(gLangID, status, ref msg);
Display("SSCPx_function", msg, status);

```

where gLangID is the id of the language, which could be set up during initialisation by the following row:

```

gLangID = (ushort)(System.Globalisation.CultureInfo.CurrentCulture.LCID & 0xFFFF);

```

6) Security settings: it might appear, that some additional security functionality would need to be added. In this case there are two ways (both are non-contradictive, i.e. it is possible to apply them both at the same time) to implement it.

The first method works on general SSCP level, i.e. is used for message encoding. It uses pair of RSA keys - private and public - and encodes given message using private key. A signature according to Public Key Cryptography Standards (PKCS#1) v 1.5 is calculated. The keys are loaded directly into MifareGlobal library object. There are six functions, which are used for this purposes:

```

public static extern UInt16 SSCP_RSA_GenerateKeyPair(UInt16 keySize, string
pubKey, string privKey); // generates key pair

```

```

public static extern UInt16 SSCP_RSA_LoadKeys(string pubKey, string privKey); //
loads keys into the library object
public static extern UInt16 SSCP_RSA_Sign_PKCS_V15(string msg, byte hashType,
string Sign); // computes compliant PKCS#1 v1.5 signature
public static extern UInt16 SSCP_RSA_Verify_PKCS_V15(string msg, string sign, byte
hashType, ref Boolean Verified); // verifies compliant PKCS#1 v1.5 signature using
public and private keys
public static extern UInt16 SSCP_RSA_Encrypt(string msg, Boolean b_256, string
Enc); // encrypts message using RSA
public static extern UInt16 SSCP_RSA_Decrypt(string Enc, Boolean b_256, string
Dec); // decrypts message using RSA

```

The second method ensures security within the host-reader communication. The functionality used are implemented on reader's level (SSCP library section) and is represented by 4 functions:

```

public static extern UInt16 SSCP_Authenticate(byte mode); //authentication with
selected authentication settings (indexed/generated key usage)
public static extern UInt16 SSCP_ResetAuthenticate(); //resets the authentication
public static extern UInt16 SSCP_ChangeReaderKeys(byte onlySoftKeys, byte
keymode, [In] byte[] SignKey, [In] byte[] CipherKey ); //change user keys
public static extern UInt16 SSCP_ConfAuthenticate(byte Mode, UInt16 Value); //
configure authentication mode, i.e. maximum number of operations before re-
authentication, maximum time, etc.

```

7) Connection properties: the connection setting by default are set to automatic, which means that the script automatically searches for a connected reader, performs connection and sets default baudrate. Nevertheless, due to unstable connectivity drivers (as mentioned before, USB connection, for example, is generally not implemented, so Virtual COM drivers are to be used), it might appear that the automatic mode will fail to perform connection or to find the reader, so those actions should be performed manually.

The functions used in this case are situated on a basic library level (SSCP). However, not all of them are useful. The manual connection function, where input is timeout, baud rate and COM port number, will look the exact way:

```

public void on(byte baudrate, int[] timeouts, string COMPort){
    SSCPlibMIFARE.SSCP_Initialize();
}

```

```

        SSCPLibMIFARE.SSCP_SetCOMType(0x01); // the port type 0x01 is COM485; 0x00
is RS232
        SSCPLibMIFARE.SSCP_Serial_SetPort(COMPort); // sets the port to exact
address
        SSCPLibMIFARE.SSCP_Serial_SetBaudRate(2); // default baudrate is 38400,
which is 2
        SSCPLibMIFARE.SSCP_Serial_SetTimeout(timeouts[0], timeouts[1], timeouts[2],
timeouts[3]); // first value is constant for read operations time-out, second -
multiplier, third and fourth - same, but for write operations
        SSCPLibMIFARE.SSCP_Connect(); // manual connection start
    }

```

Default initialisation (i.e. the one using AutoConnect, which is enabled by default) should however work fine.

8) GUI description: the graphical user interface for the application should be simple and consist of several buttons which correspond to the functionality of the tool. The most minimalistic variant contains no text fields, and could be seen in Figure 10.

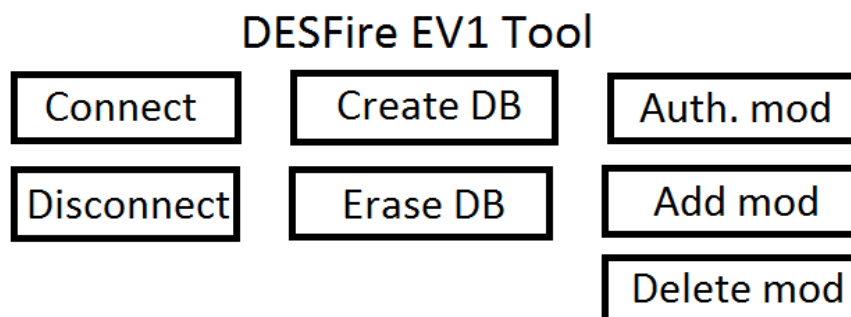


Figure 10: Basic GUI

However, considering the fact that the system user will probably want to know whether the operation has completed successfully, and see some operation data, a text label, which will display last operation's result should be added. In case the user will want to save the session log, a button which implements that operation should also be present on the GUI.

The code will then be changed in the following manner: as it was mentioned previously, every SSCP method returns two bytes of status telling whether the operation has been completed successfully or not. Those statuses shall be sent both to the label and a stack, which later on will be used for generating the log. The following code should be added before the class body:

```

using System.Drawing;
using System.Windows;
using System.IO;

```

And this is what has to be added directly to the class body:

```

StreamWriter w = File.AppendText("log.txt");
bool log = true;

public static void Log(String Command, String Result, uint status)
{
    w.Write("\r\nLog Entry : ");
    w.WriteLine("{0} {1}", DateTime.Now.ToLongTimeString(),
        DateTime.Now.ToLongDateString());
    w.WriteLine("  :");
    w.WriteLine("  :{0}", "Command : " + command + ", result : " + result + ",
status : " + status + "");
    w.WriteLine ("-----");
}

private void Display(String Command, String Result, uint status) //displaying
operation result
{
    if ((byte)status == SSCPLibMIFARE.SSCP_OK)
        { // in case of success
            labelResult.ForeColor = Color.Green;
            labelStatus.ForeColor = Color.Green;
        } else
        {
            labelResult.ForeColor = Color.Red;
            labelStatus.ForeColor = Color.Red;
        }
    labelResult.Text = Result;
    labelCommand.Text = Command;
    labelStatus.Text = "Status word :" + status.ToString("X4"); //
hexadecimal format
}

```

```
private void CommandInProgress() //displaying running command status
{
    labelCommand.ForeColor = Color.Blue;
    labelCommand.Text = "Command in progress...";
    labelResult.Text = "";
    labelStatus.Text = "";
}
```

where labelCommand is a text label, representing actual command launched, labelResult holds the result of the command, and labelStatus represents status of the answer received from the reader. Those variables need to be defined in Designer class of the application.

Now any SSCP operation shall be surrounded by the following template:

```
CommandInProgress();
UInt16 status = SSCPx_function(arg a1, ...);
if ((byte)status == SSCPLibMIFARE.SSCP_OK)
{
    mesg = "Success"; // the text should be changed according to the
operation
}
else SSCPLibMIFARE.SSCP_GetErrorMsg(gLangID, status, ref mesg);
Display("SSCPx_function", mesg, status);
if(log){
    Log("SSCPx_function", mesg, status);
}
```

The updated interface will now have the following layout (Figure 11)

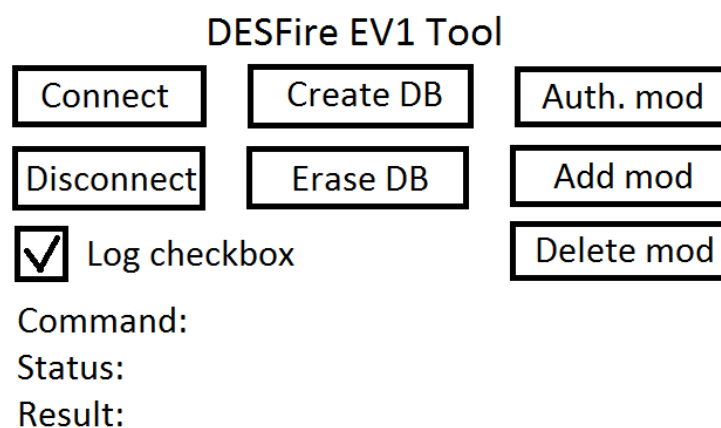


Figure 11: Advanced GUI

But there is also one more case which needs to be considered: it is when automatic connection fails, and the user should manually set up the connection. For this situation, several text fields should be provided, where the user enters connection data. Also a checkbox which defines which initialisation type will be used must be added.

The back code changes in this case will touch on() method only.

```
public void on(){
    SSCPLibMIFARE.SSCP_Initialize();
    SSCPLibMIFARE.SSCP_SetMode(1);
    if(manual.Checked){
        SSCPLibMIFARE.SSCP_SetAutoConnect(0x00);
        (
        byte b =
        (String.Equals(portType.SelectedValue.ToString(),"RS485",StringCompar
        ison.Ordinal)) ? 0x01 : 0x00;
        SSCPLibMIFARE.SSCP_SetCOMType(b); // the port type 0x01 is COM485;
        0x00 is RS232
        SSCPLibMIFARE.SSCP_Serial_SetPort(port.SelectedValue.ToString()); //
        sets the port to the value selected from the dropdown list
        SSCPLibMIFARE.SSCP_Serial_SetBaudRate(baudRate.SelectedValue.ToString
        ()); // default baudrate is 38400, which is 2
        SSCPLibMIFARE.SSCP_Serial_SetTimeout(Int32.Parse(readTOC.Text),
        Int32.Parse(readTOM.Text), Int32.Parse(writeTOC.Text),
        Int32.Parse(writeTOM.Text)); // first value is constant for read
        operations time-out, second - multiplier, third and fourth - same,
        but for write operations
        SSCPLibMIFARE.SSCP_Connect(); // manual connection start
    }
}
```

The alternative interface with manual connection settings options should look like the one shown in Figure 12.

DESFire EV1 Tool

Connect	Create DB	Auth. mod
Disconnect	Erase DB	Add mod
<input checked="" type="checkbox"/> Manual connection	Delete mod	

Port: Timeouts

Port type: Read Const:

BaudRate: Read Multi:

Write Const:

Write Multi:

Log checkbox

Command:

Status:

Result:

Figure 12: Advanced GUI with manual connection

From the general point of view, that is all. Perhaps in case of a more detailed proposal other additions would be necessary. Also the fact that this code is only a prototype shouldn't be forgotten. But even this should be enough to ensure basic functionality of the system from the host's side.

4. Results

In this section, the final system proposal is analysed. In the first part, drawbacks and advantages would be listed and weighed. In its second part, alternative system architectures will be compared to the solution. Finally, general applicability of the system and the project in general will be evaluated.

4.1 Final proposal analysis

The final solution, which is proposed, considers using more elements than it was planned before. Mainly this is a consequence of the technical limitation, which come together with the STid ARC-A reader itself and Mifare Global software functionality.

It uses data storage of the reader, which brings some localisation of the authorisation data. This may both be useful, as the data will not be centralized, so in case the main server data will become vulnerable, no sensitive data will appear in threatening situation. On the other hand, keeping stored data, even though they are encoded, on an external ROM (i.e. reader itself) also appear to be not the safest solution ever. Nevertheless, the scenario of stealing and decoding the whole reader seems to be fairly unlikely.

The host PC plays in the architecture much more significant role. It is there where the script runs, though at the beginning of the project the main idea was to make the code directly for the reader. The main problem though is that such “programmable” readers aren’t widespread and have dramatically higher price. Moreover, the functionality of “Match-on-card” biometrics in those readers, which support it, is natively implemented, so there would be either no need or no possibility to write the code directly for them, as former case requires no code advance at all, and latter one still has its functionality controlled directly by the host PC.

Coding for Mifare reader is challenging in some aspects, mainly concerning data types used, byte operations, operations sequence and error control. Luckily the wrappers have all the functions implemented, so there was no need to double-check the datatype correspondence. However it needs to be mentioned that documentation has several sufficient drawbacks, which lead to, for example, misunderstanding of how to use automatised mode.

- Inability to use full-text search through the document
- Not all the functions explained as widely, as they should have been
- Lack of examples in some cases
- Some of the types used were not mentioned at the beginning

- Several mistakes found (not critical though)

Otherwise there are no problems with the documentation file. Mifare Global library, despite lack of some demanded functions (for example, retrieving or checking exact UID presence in the database or retrieving fingerprints associated with the UID), has basically all the necessary functionality to work with the reader implemented in it. It is understandable thought, as retrieving fingerprint templates directly from the reader, which makes such functions a potential security threat to the system in general, co implementing standard CRUD operation set seems senseless. Unfortunately, due to the limitations, user enrolment into the system has to be done manually (with administrator's participation), same as removal of a record from the database (which could have been also done automatically). Nevertheless, assigning or removing several records in a row should not require more effort than removing a single record, so from this point of view it looks more positive and practically useful.

Implementation of the GUI and testing, taking into consideration small amount of code, narrow specialisation of the software and no demand on using high-end graphics, should not take very long.

In general, the system can be taken as an alternative for the existing system. The question, however, remains, how useful it would be, especially in comparison with the alternatives, and is it worth considering such improvement for the security system of CTU.

4.2 Comparison with the alternatives

The system already was in some way compared with the alternative solutions, mainly in security, performance and architectural aspects. This was, however, done before the code implementation, i.e. before the functionality of the system is known and somehow implemented. The situation has changed, and so the comparison may now include such aspect as maintenance and implementation.

First of all, the alternatives should be reminded. Those are:

- 1) "Match-on-card" biometrics - the reader has built-in support of direct comparison of the card-stored fingerprint and live fingerprint templates
- 2) Host-side solution - the templates are stored on the host PC instead of the fingerprint reader

- 3) Server-side solution - all the templates are stored on one central server together with other user data

“Match-on-card” solution, as it has been described before, can also have two variants: either fully-autonomous reader, which runs code by itself, and requires no host PC, or the reader which though has the feature of reading on-card templates, but the code runs on the host. The former solution requires less elements to be used in the architecture, but such readers definitely cost significantly more, and programming languages used there are restricted (probably Java, as it is the most popular multi-platform language, or even some lower-level language). Usage of such solution for the CTU is questionable. The latter variant might be a solid solution, as it has major similarities with this project’s proposal. However, price issues need to be taken into consideration, as well as need to store the data on the card itself (possible memory issues). The coding in this case will have approximately on the same level or slightly less hard.

Host-side solution will not require usage of reader’s internal memory. Code structure in this case remains close to the original. The search function should be implemented inside the program. It is questionable, however, whether the host PC is a better storage then the reader itself. Practice shows that PC and servers are more vulnerable to different threats: the theft, who has gained access to the computer, can simply clone or modify the local database (while with the reader such scenario is much less possible) with some knowledge of programming. While, on the other side, there is no direct command for the Mifare Global, which allows to retrieve fingerprints from the reader. But this also brings some backup issues, i.e. there is no possibility to make a backup, and in case of the electrical hazard the data loss will be irretrievable, and all the users of the system need to be enrolled again. The following fact needs to be mentioned: there is no direct need to use the card in this case. However, removing one more security measure, though such unreliable, is not necessary.

Server-side solution, except from data centralisation (also questionable, as the only advantage it has is easier access to the data by administrator’s side), has no preferences over the current architecture. First, the data need to be sent directly to the server, so more network coding will be required. Second - increased security threats due to the centralisation of the data (access to all data at once). Third - technical threats (in case of server hazard, all data will be lost, in case of connection problems all locations will be blocked). This would also require restructuring CTU’s access architecture in general, so this solution is definitely undesirable.

There is one more option, which was not mentioned before due to the lack of information - Javacard. This is a technology, which allows writing the script among the data directly on the card, so the reader is used only as a JVM which runs it. This technology opens a possibility to directly encode "Match-on-card" biometrics even without the basic support of it from the reader. However, Javacard is another card type, so accepting this solution would require not only changing all (without exception) the readers at CTU, but also changing all the cards among the CTU staff and students. This would lead to such significant cost increase, that this architecture is clearly undesirable.

The conclusion for this part is the following: only one architecture ("Match-on-card" reader with host (ex. "STid ARC-D") may (but not necessarily must) be more optimal. In order to decide, whether it is worth considering this solution, more market offers on this product should be studied through. Otherwise, the one offered in this project remains the most optimal in all the aspects. It is easy to implement, cheaper than the alternatives, secure, independent on other similar systems.

The only aspect which could be thought through is using only fingerprint scanner (without RFID card), so that live fingerprint would be directly searched through the database. However, this is not generally necessary.

4.3 Usage of the project

This project aimed to explore the possibility of introducing biometrics to the CTU's systems. From this point of view, the information from this work may be useful for those, who:

- Have a general interest in biometrics, its history and modern methods
- Have specific interest in usage options of modern biometrical method in security purposes and more technical view on the problematics
- Deal with security systems on daily basis, specialise on developing/modifying security systems and architectures
- System engineers
- CTU staff
- CIPS workers
- Programmers with special aim on card systems

How the project may be used in future:

- Demonstrating system examples and comparison of the biometrics+RFID systems
- Developing new systems in CTU
- Studying possibilities of Mifare DESFire technology

- Preparing studying material on biometrics
- Starting implementation of real-life project on biometrics application

The code from this project, despite being only example, should grant a solid basis for a fully-functioning program for working with Mifare DESFire cards and fingerprint readers.

4.4 Conclusion

In this this project a quick look on biometrics, its applications and usage in our everyday life has been granted. Moreover, some advanced information on the technological aspects has also been introduced. The analysis of possible solutions consisted of reviewing the alternatives, selecting the one according to several aspects, such as complexity, usability, price, technological limitations and current system working in CTU. The technical solution description consisted of presenting technical instruments (software and hardware), system engineering using UML diagrams and stating final architecture of the system. The development of the applet's code included pre-analysis: functional requirements, software development kit description, its features and limitation analysis. Then the prepared code was described and analysed according to the possible threats, usage issues and future development, where possible solutions have been also presented. In the end you will find information on the whole system's usability and its correspondence with the task, end comparison of the solution in accordance with the experience gained while developing the code, and summarization of system's and project's value and usability in the future. The only possible thing lacking in the project is testing the code, which, however, still needs to be done during future GUI development.

The task of the project may be considered as fulfilled.

List of attachments

1. Code of the applet
2. Mifare Global manual
3. Mifare Global function wrapper to C#
4. Mifare Global DLL library

List of abbreviations

AES – Advanced Encryption Standard
AFIS – Automated Fingerprint Information System
APDU – Application Protocol Data Unit
ARAT – Active Reader Active Tag system
ARPT – Active Reader Passive Tag system
CIPS – [cz] Centrum Informačních a Poradenských Služeb
CMAC – Cipher-based Message Authentication Code
COM – Communication Port
CPS – [fr] Cartes de professionnels de santé
CRUD – Create, Read, Update, Delete
CTU – Czech Technical University in Prague
DB - Database
DC – Direct Current
DES – Data Encryption Standard
DLL – Dynamic-link Library
DNA – Deoxyribonucleic Acid
EAL – Evaluation Assurance Level
EEPROM – Electrically Erasable Programmable Read-Only Memory
FBI – Federal Bureau of Investigation
FBMI – Faculty of Biomedical Engineering
GUI – Graphical User Interface
ID – Identification (number)
IP - International Protection Marking
ISIC – International Student Identification Card
ISO - International Organization for Standardization
LED – Light-Emitting Diode
MAD – Mifare Application Directory
PIN – Personal Identification Number
PC – Personal Computer
POR – Power-On-Reset
PRAT – Passive Reader Active Tag system
RF – Radio Frequency
RFID – RF Identification
RSA – Rivest, Shamir, Adleman (authors names)
SDK – Software Development Kit
SMS – Short Message Service
SSCP – Secure STid Connection Protocol
TIFF – Tagged Image File Format
UID – Unique ID
UML – Unified Modelling Language
USB – Universal Serial Bus

List of figures

Figure 1: Human skin structure	19
Figure 2: Fingerprint patterns [2]	20
Figure 3: Ridge ending, bifurcation and short ridge (dot) [4].....	21
Figure 4: Fingerprint core and delta location [5]	21
Figure 5: System component diagram.....	30
Figure 6: Main activity (authentication) diagram	31
Figure 7: SESPro interface	32
Figure 8: SESPro Mifare DESFire EV1 settings	33
Figure 9: DESFire EV1 card structure	34
Figure 10: Basic GUI.....	50
Figure 11: Advanced GUI.....	53
Figure 12: Advanced GUI with manual connection.....	54

List of tables

Table 1: Operational criteria of biometric methods [2]	17
Table 2: General comparison of 6 most popular methods [3]	18
Table 3: System elements.....	30
Table 4: Command APDU structure	34
Table 5: Data formats [Att. 2]	37

Literature and references

- [1] Jain, L.C. et al. (Eds.). 1999. *Intelligent Biometric Techniques in Fingerprint and Face Recognition*. Boca Raton, FL: CRC Press.
- [2] RAK, Roman, Vašek MATYÁŠ a Zdeněk ŘÍHA. *Biometrie a identita člověka ve forezních a komerčních aplikacích*. Praha: Grada, 2008. Profesionál. ISBN 978-80-247-2365-5.
- [3] BY RUUD M. BOLLE .. [ET AL.]. *Guide to biometrics*. New York, NY: Springer, 2004. ISBN 9781475740363.
- [4] <https://inotes4you.com/2013/09/12/>
- [5] http://anilaggrawal.com/ij/vol_002_no_001/papers/paper005.html
- [6] Kitsos, PARIS, and Yan ZHANG. *RFID Security: Techniques, Protocols and System-on-chip Design*. New York: Springer, 2008. Print. ISBN 978-0-387-76481-8
- [7] <https://www.hidglobal.com/products/readers/iclass/rw100>
- [8] http://www.nxp.com/products/identification-and-security/smart-card-ics/mifare-ics/mifare-desfire:MC_53450
- [9] <https://www.mifare.net/wp-content/uploads/2015/04/MIFARE-DESFire-EV1-Registered.pdf>
- [10] http://www.nxp.com/documents/application_note/AN10787.pdf
- [11] https://www.blackhat.com/presentations/bh-usa-08/Buetler/BH_US_08_Buetler_SmartCard_APDU_Analysis_V1_0_2.pdf
- [12] http://stid.com/catalog/index.php?id_product=127&controller=product&id_lang=2
- [13] <http://web.archive.org/web/20090630004017/http://cheef.ru/docs/HowTo/APDU.info>