SOFIA UNIVERSITY
"ST. KLIMENT OHRIDSKI"
FACULTY OF MATHEMATICS AND INFORMATICS
DEPARTMENT OF SOFTWARE TECHNOLOGIES

Securing Data during Storage, Management and Transfer between Mobile Devices

by

Peter Pashinov Sabev

ABSTRACT

for a thesis submitted in fulfillment of the requirements for the educational and scientific degree "Doctor"

in

Professional field: 4.6 "Informatics and Computer Science",
Doctoral program: "Software Technologies" – Software Engineering

Scientific advisor:
Prof. Milen Yordanov Petrov, Ph.D.

Sofia,
2024

Contents

Chapter 1: Introduction

1.1. Motivation and Relevance of the Topic

Nowadays, mobile devices and technologies are gaining more and more importance in people's every day life. Thanks to the significant evolution of mobile devices in recent years, expressed both in a significant improvement in their computing and communication capabilities, and in a significant improvement in their capabilities to store and process diverse types of data, including biometric and personal / sensitive data, the mobile devices are becoming more and more popular for performing a great variety of tasks in many fields, having efficiency comparable to the efficiency of desktop machines. Mobile devices and technologies applicability in fields like mobile payments, banking, submission of applications, health information, legal information, financial data and other types of data and official documents to governmental / private / other institutions in combination with the typical use of mobile devices for private correspondence, requires the entry, processing, storage and transfer of many personal and/or sensitive data, for which data it is crucial to be ensured adequate protection, as evidenced by the general and specialized legislative initiatives on a global scale, related to data protection. Moreover, mobile devices distinctive characteristics like good portability and always-on nature, impose additional requirements from security standpoint. This all gives rise to the need both to conduct studies, analysis and security evaluation of mobile devices software in terms of data security effectiveness, and to develop specialized approaches, methods and architectural solutions for the design and development of highly secure software (HSS) that is able to ensure adequate data protection for their full lifecycle, including while the data is stored on the device, is being processed, managed or transferred between different mobile devices.

For any software, regardless of its type, purpose and application domain, it is crucial to implement a secure authentication mechanism in order to prevent illegal access to its data. Many authentication mechanisms have been developed over the years, with a number of studies being conducted for improving existing authentication methods and developing a new ones. However, the password-based authentication method still remains the most widely used and preferred method to date, with indications that its dominant position will be preserved in the near future. [1]–[5]. This is a result not only of the large number of existing legacy systems and applications, implementing password-based authentication, but also of the advantages coming with the method (advantages like a good balance of simplicity, usability and security). However, there are also some disadvantages coming with the password-based authentication that are related primarily to the human factor and the limitations of human memory, as evident from studies [6] and [7]. All this gives rise to the need for a special-purpose software called password management software (PMS), having the main responsibility to improve the overall security of password-based authentication process by automating the secure passwords creation, management, storage, retrieval and transfer process in a way making it possible to be achieved a maximal level of security during all phases of passwords lifecycle. This need is also evidenced by a report, pointing out that 81% of registered data security breaches are due to stolen / weak passwords [8]. In addition, the increasingly important role of PMS in terms of data security and protection is also confirmed by the security experts recommendations about using PMS [9]. Related to this, it is important to be noted that this role imposes particularly stringent security requirements on the PMS, especially if compared to the standard general-purpose software. All this, combined together with the ever-increasing popularity of PMS, evident from studies like [5], makes the following question particularly relevant: "To what extent do the representatives of the existing PMS is actually able to meet the high expectations for this type of software in terms of security?", because PMS's design goal to be able to provide highly secure storage for security credentials for a large number of third-party systems and/or applications, makes the PMS a particularly attractive, high-priority target for attacks.

Given the people's needs to securely store and manage a great variety of sensitive data, in addition to their passwords, the classic PMS is evolving into a special-purpose software known as a digital vault software (DVS). Moreover, this evolution poses even greater security challenges compared to the classic PMS, because for the DVS sensitive data is not only limited to passwords / security credentials. The DVS should be able to be able to provide a reliable and secure long-term storage, processing and management for sensitive data of any kind while continuing to provide at least as high a level of security as the PMS.

The dissertation is primarily focused on mobile devices HSS (PMS / DVS primarily) related researches, including HSS security analysis, security evaluation (including data security effectiveness evaluation), approaches, methods and architectural solutions for data security allowing for the design and development of secure and reliable PMS / DVS. More specifically, the research made as part of the dissertation, including the studies and experiments conducted, are primarily focused on PMS / DVS for mobile devices running the operating system with the largest market share at the moment, namely *Android* [10], but they are also applicable to any other type of HSS for which data security and protection are of critical significance and importance, regardless of the concrete domain, application area and/or operating system.

1.2. Object and Subject of the Study

The highly secure software, designed to provide secure data storage, processing, management and (eventually) transfer between mobile devices, is the study's main object with the main focus placed (due to the ever-increasing need for PMS / DVS software, including the importance to be evaluated what actually has been achieved in the field so far in terms of security) on security research, security analysis and security evaluation of PMS / DVS, the design and implementation of which poses one of the biggest challenges in terms of security, given that any eventual mistake, even the smallest one, may prove fatal not only for the security of the data itself, but also for the security of all third-party systems / applications for which the data are intended to be used to make possible completion of security-critical tasks like identity verification, remote payments and many others.

The main subject of the study is to conduct a digital investigation, including analysis and security evaluation of PMS / DVS for mobile devices, together with a research focused on significant methods, approaches, primitives, standards, tools of cryptography, guidelines and design patterns for building secure software allowing to be defined methodologies for analyzing and security evaluating PMS / DVS both in a generalized (in breadth, at high level) and specialized (in depth) manner (in terms of one of the most important security aspects, namely the security aspect about securing the software's data in its own main memory areas, as well as data exposure limitation).

1.3. Dissertation Aim and Objectives

The main aim of the dissertation is to support the security research of PMS / DVS, including to security analyze and security evaluate this type of highly secure software both generalized (in breadth, at high level) and with respect to the data security and protection effectiveness while PMS / DVS is running into the main memory, including the security measures the PMS / DVS is taking in order to limit sensitive data exposure in its own main memory areas.

In this regard, the following research objectives are outlined:

1. Make a general overview (in breadth) in the researched field, by:

- **1.1.** Making an overview in the field of data security and main data security threats identification, on basis of which to formulate a fundamental security requirement for data protection in context of a HSS;
- **1.2.** Researching for guidelines and recommendations for building secure software, including design patterns, approaches and techniques for securing data during storage, management and transfer between mobile devices;
- **1.3.** Researching for existing specialized tools in the field of digital forensics, security analysis and vulnerabilities identification;
- **1.4.** Making an overview (in breadth) in the field of in-memory data protection, reviewing and analyzing the state-of-the-art approaches related;

2. Define a methodology for conducting an investigation, comparative security analysis and in breadth security evaluation of HSS at a high-level;

3. Validate the applicability of methodology (defined in point 2) by applying it to conduct an investigation, comparative security analysis and in breadth security evaluation of a representative sample of PMS / DVS. For this purpose:

3.1. Analyze at high-level the publicly available official information, including the technical documentations available and the independent security researchers results, for at least 4 popular Android applications defining themselves as PMS / DVS in order to form a representative sample of at least two Android applications for future study.

3.2. Conduct an investigation, including do the experiments necessary in order to security analyze and in breadth security evaluate at high-level the applications from the representative sample formed in point 3.1;

3.3. Prepare a high-level comparative security analysis and security evaluation report for the applications from the representative sample, containing also an information about the studied applications security awareness and transparency degree based on the publicly available sources (official documentations, design and implementation notes, etc.);

4. Define a main memory inspection methodology with regard to the efficiency of in-memory data protection in the context of HSS execution;

5. Design, describe and implement, a specialized software tool for identifying and extracting exposed unprotected trusted data in main memory in order to support the conduct of digital investigation, analysis and security evaluation regarding the effectiveness of the protection of data entrusted to a given HSS in main memory areas belonging to it;

6. Validate the applicability of both the methodology defined in point 5 and the specialized software tool developed in point 6, by applying them to conduct a digital investigation to security evaluate the effectiveness of in-memory data protection in the context of representative sample HSS execution. For this purpose:

6.1. Select a representative sample of two applications based on the results from point 3.3;

6.2. Conduct a digital investigation in order to security evaluate the effectiveness of in-memory data protection in the context of execution of specific HSS from the representative sample (from point 6.1);

6.3. Analyze the digital investigation results (from point 6.2), on the basis of which prepare a comparative security analysis for the representative sample HSS;

6.4. Prepare a security assessment report for the effectiveness of in-memory data protection in the context of execution of specific HSS from the representative sample (from point 6.1);

1.4. Applicability and Expected Benefits

The research, experiments and results in the dissertation, on the basis of which it is planned to prepare both a generalized comparative security analysis with a security assessment (in breadth) of a concrete existing PMS / DVS, and a specialized comparative security analysis with an assessment of security (in-depth) regarding one of the most important aspects in terms of security, namely the security aspect about securing the software's data in its own main memory areas and limitation of data exposure, are expected to find numerous applications to support organizations / institutions (private or public) / persons interested in this type of software (from ordinary mass users to specialized researchers and developers in the field) in a number of application areas, among which:

- Making a reasoned and informed choice of specific existing PMS / DVS, especially when it is needed to make the choice decision on the basis of specific security criteria, the satisfaction on which is expected to be supported by concrete facts, based on specific studies and experiments that are able to reveal valuable information about the studied software behavior in certain risk situations;
- Identification of possible weaknesses in existing HSS of type PMS / DVS, including identification of deficiencies and shortcomings from security point of view in relation to already developed / chosen approaches and architectural solutions for data security and protection;
- Rationale preparation regarding the need to design and implement new HSS of type PMS / DVS in order to be met the needs of a given organization, / institution / stakeholder requiring satisfaction of certain security criteria;

In addition to the above benefits and application areas, the methodologies defined in the dissertation and all related definitions in terms of security evaluation criteria, specialized usage

scenarios, synthetic trusted data sets, specialized software tool for identifying and extracting exposed unprotected trusted data in main memory, etc., are expected to serve as a framework that is able to contribute to assisting the researchers and the developers in the filed regarding:

- Independent digital investigation conduction for any existing software, including comparative security analysis with a security assessment report preparation for the software studied;
- The design and implementation of new approaches, methods, tools, techniques and architectural solutions for designing and implementing new HSS, including for improving an already existing HSS, with an option for the achievements to be measured empirically from security standpoint based on concrete comparative security analysis and security evaluation results;

1.5. Dissertation Structure

The dissertation consist of five chapters. It's main text is written in 168 pages, supplemented by 44 figures, 8 tables and 3 appendices, representing both graphical and textual the results of the studies and experiments performed. The total number of cited literature sources is 289, including citations to scientific work of independent security researchers, books, standards, websites and other sources. The dissertation is having the following structural organization:

Chapter 1: The purpose of this chapter is to serve as an introduction, representing in a concise form the motivation behind the dissertation's main goal, including the expected benefits of its implementation. Structurally this chapter consists of the following 5 semantically distinct sections:

Section 1.1: This section describes the motivation and relevance of the topic of the dissertation work;

Section 1.2: This section describes the object and subject of the study for the dissertation work;

Section 1.3: This section describes the main goal of the dissertation work and sets the tasks resulting from this;

Section 1.4: This section describes the expected benefits of the fulfillment of dissertation's main goal and resulting tasks set;

Section 1.5: This section describes the structural organization of the content of the dissertation;

Chapter 2: This chapter provides a summary of the general overview (in breadth) in the researched field made as part of the dissertation work, together with the accompanying analysis, on the basis of which the main definitions and formulations are made with the goal to support the planned studies and experiments. Structurally this chapter consists of the following 5 semantically distinct sections:

Section 2.1: This section is composed of 3 sections describing the general overview made as part of the dissertation work in the field of data security, security requirements engineering and data security threats identification; It also formulates a fundamental security requirement for data protection;

Section 2.2: This section is composed of 6 sections describing the research made as part of the dissertation regarding guidelines and recommendations for building HSS, including design patterns, approaches and techniques for securing data during storage, management and transfer between mobile devices;

Section 2.3: This section briefly describes the main capabilities of some of the tools selected during the research done as part of the dissertation work to support the planned digital investigations, security analysis and vulnerabilities identification;

Section 2.4: This section summarizes the Chapter 2 performed activities, together with the main conclusions made with regard to the studies described in the chapter;

Chapter 3: This chapter describes the high-level study of HSS made as part of the dissertation work, together with the comparative security analysis and in breadth security evaluation of a concrete PMS / DVS made on the basis of concrete security criteria defined. Structurally this chapter consists of the following 3 semantically distinct sections:

Section 3.1: This section describes the definition of a methodology for conducting an investigation, comparative security analysis and in breadth security evaluation of a HSS at high-level;

Section 3.2: This section consists of 3 sections describing the process of validating the applicability of the methodology defined in Section 3.1; It shows that the methodology can be successfully applied to conduct an investigation, comparative security analysis and in breadth security evaluation of a concrete HSS of type PMS / DVS at high-level;

Section 3.3: This section summarizes the Chapter 3 performed activities, together with the main conclusions made regarding the studies described in the chapter;

Chapter 4: This chapter describes the research carried out as part of the dissertation work in regard to in-memory data protection in the context of a concrete HSS execution, including the results of the performed digital investigation together with the experiments made, the comparative security analysis and the security evaluation of the effectiveness of protection of the data entrusted to the examined software in main memory areas owned by it. Structurally this chapter consists of the following 4 semantically distinct sections:

Section 4.1: This section describes the definition of a main memory inspection and in-memory data protection effectiveness measurement methodology in the context of HSS execution;

Section 4.2: This section consists of 4 sections describing the developed as part of the dissertation work specialized software tool for identifying and extracting exposed unprotected trusted data in main memory in order to support the conduct of digital investigation, analysis and security evaluation in regard to protection effectiveness of the data entrusted to a given HSS in main memory areas belonging to it;

Section 4.3: This section is composed of 6 sections, in which the applicability of both the methodology defined in Section 4.1 and the developed software tool described in Section 4.2 is validated by applying them to conduct a digital investigation in order to security evaluate the effectiveness of in-memory data protection in the context of representative sample PMS / DVS execution;

Section 4.4: This section summarizes the Chapter 4 performed activities, together with the main conclusions made regarding the digital investigation described in the chapter, the experiments and the security assessment made;

Chapter 5: The purpose of this chapter is to serve as a conclusion that summarizes the main activities carried out to achieve the dissertation's main goal and to implement the tasks resulting from it. In addition, this chapter describes the main contributions of the dissertation, including the publications and reports related to it and the directions for future research. Structurally this chapter consists of the following 4 semantically distinct sections:

Section 5.1: This section summarizes the fulfillment of dissertation's main goal and resulting tasks set;

Section 5.2: This section describes the main contributions of the dissertation work;

Section 5.3: This section describes the main publications and reports related to the topic of the dissertation;

Section 5.4: This section describes the directions for future research;

Chapter 2: Current State in the Research Field

This chapter provides a summary of the general overview (in breadth) in the researched field made as part of the dissertation work, together with the accompanying analysis, on the basis of which the main definitions and formulations are made with the goal to support the studies and experiments planned.

2.1. Data Security, Security Requirements Engineering and Data Security Threats Identification

This section provides a general overview in the field of data security, security requirements engineering and data security threats identification, introducing key terms and definitions related to data security. In addition, the main data security threats are formulated, which tends to be some of the most significant ones regarding data security in the context of HSS. Subsequently, a fundamental security requirement for data protection in the context of HSS is also derived.

2.1.1. General Overview in the Field

The results analyzed for *"data security"* keywords based search query to the global abstract and citation database of peer-reviewed literature *Scopus* are showing that the number of scientific publications related to data security is increasing every year [11]. A similar trend is also observed with regard to the topics that are more closely related to mobile applications security requirements in the context of data security, as well as to the topics on data security threats, which are related to the security threats for data used somehow by a certain mobile device software. All this, together with the significant count of publications on the these topics for the period observed (2012 to 2022) is an evidence confirming the increasing interest of the scientific community on the topics over the years.

In the context of HSS and especially in the context of a specialized HSS like PMS / DVS it is very important to be ensured as strong data security as possible. Related to this, a number of security requirements must be met. Security requirements engineering is the process of derivation and specification of security requirements helping to be ensured that the software that is being build will be secure and reliable enough to meet the stakeholders needs, including that this software will be able to continue its operation correctly even in the event of errors, malicious behavior, intentional attacks, and random incidents or accidents [12], [13]. Essential to the security requirements elicitation and specification process is to be both identified and analyzed carefully the potential security threats in the context of the software that is being developed, including the potential weaknesses and vulnerabilities that may be exploitable in such a way that will allow a potential attacker to breach the security. Also, the engineers involved in this process must have the ability to think like attackers, with all the technical skill and knowledge necessary, including the skill to identify potential attack vectors. The literature review made as part of this section surveys prior research and based on this provides an information on the existence of a variety of approaches, method and methodologies about potential security threats identification and analysis, including about security requirements engineering. It also shows that both a great variety of security threats may be identified and a great variety of security requirements may be elicited and specified depending on the stakeholders goals and needs.

2.1.2. Main Data Security Threats in the Context of Highly Secure Software

The main data security threats (ЗСД) that tends to be one of the most significant to the security of the data in the context of a PMS / DVS and in general for a HSS of any type are described briefly bellow:

- **ЗСД_№1 : Sensitive data exposure** – Sensitive data entrusted to a HSS are considered exposed when the confidentiality of the data is compromised.
- **ЗСД_№2 : Data** – Data entrusted to a HSS are considered tampered with when the authenticity / integrity of the data is compromised.
- **ЗСД_№3 : Security settings tampering** – Security settings of a HSS are considered tampered with when their authenticity / integrity is compromised.

In order to support the above-formulated data security threats and to support the subsequent formulations / definitions, including the experiments and the research work planned, the dissertation provides definitions for the following terms and concepts: User-entered data entrusted to the HSS (denoted as **trusted data** or **ДД**); **Raw value** (**СС**); **Metadata**; **Security settings**; **Security credentials** (**ТУУ**); **Confidentiality**; **Authenticity**; **Integrity**;

2.1.3. Fundamental Security Requirement for Data Protection in the Context of Highly Secure Software

Fundamental Security Requirement for Data Protection (ОИСЗД) : An application defining itself as a HSS should be able to provide cryptographically strong multilayered data protection ensuring at least the integrity and authenticity of both the public and sensitive data and also the confidentiality of the sensitive data. More precisely, each HSS should be built upon a strong security architecture (upon the zero-knowledge security concept) enforcing secure data processing, secure data management and secure data storage during all phases of data's lifecycle by taking into consideration the concrete specifics of the data, the device on which the HSS instance is running, including the execution environment and the potential data security threats (at least ЗСД_№1, ЗСД_№2 and ЗСД_№3). In particular, the multilayered data protection should be built on top of a security-enforcing core, including robust security mechanisms, modules, and subsystems that are able to fully meet the specified as part of the dissertation work: **ОИСЗД_№1.1**; **ОИСЗД_№1.2**; **ОИСЗД_№1.3**; **ОИСЗД_№1.3.1**; **ОИСЗД_№1.3.2**; **ОИСЗД_№1.3.3**; **ОИСЗД_№1.4**; **ОИСЗД_№1.5**;

Figure 1: Fundamental Security Requirement for Data Protection (ОИСЗД)

It is important to be noted that the Fundamental Security Requirement for Data Protection (ОИСЗД), that is shown on Figure 1, sets the lower bound for security with regard to a HSS.

2.1.4. In-memory Data Protection: Overview and State-of-the-Art Approaches

Typically, for a given software to be run, it is needed to be created a process in the context of which an instance of the software is being run and all data that are needed for its operation are being loaded in cleartext in main memory. In general, if no application level supplementary measures are taken by the concrete software instance that is running, the data loaded will continue to reside in cleartext in main memory portions belonging to the address space of the process in the context of which the software is running until security measures are taken to be ensured an adequate data protection that is based on cryptographically strong tools and techniques.

In theory, the operating system is responsible for enforcing a full isolation between the processes that are running. However, when such an isolation is applied alone, it cannot be considered enough, especially in terms of in-memory data protection, given the potential risks for the data inherent in mobile devices, for which devices there is an increased risk to be potentially lost or stolen when compared to desktop machines. This, combined with data remanence effect and the well-known challenges regarding secure deletion of data loaded in main memory, increases the chance for a successful attacks targeting to break the confidentiality of data loaded in main memory, especially given the significant progress in the development of tools and techniques to study and analyze the contents of main memory for the purpose of extracting cleartext data from it. With this in mind, the dissertation provides an example of well-known and time-proven attacks with the respective research papers showing that these attacks are both feasible and applicable for sensitive data extraction from the RAM of *Android* mobile devices, even in a case of a locked device bootloader. Also, it is noted that the feasibility of the attacks and the improvements achieved with regard to the capabilities of tools and techniques for sensitive data extraction have prompted numerous studies to be conducted in response to the identified problems regarding the security of data in RAM, aiming to develop a reliable solution to protect the data in the RAM. Attention is paid to scientific works focused on development of specialized methods and protection measures, including specialized mechanisms built on top of secure hardware features like *TrustZone*, *Knox*, *Verified Boot*, *etc*. It is concluded that although such type of specialized approaches and safeguards make data extraction from RAM extremely difficult and even overwhelming for non-resourceful adversaries, the same cannot be said for resourceful adversaries,

such as government / military / intelligence organizations / adversaries for whom it is feasible to discover and exploit "secrets" of manufacturers / vendors; to force them to "cooperate"; to effectively apply highly technical methods and techniques for the exploitation of vulnerabilities / weaknesses specific to the concrete chips / boards / components, etc. Examples of successful bypassing of standard security mechanisms of popular *Android* devices are provided.

2.2. Guidelines and Recommendations for Building Highly Secure Software

This section introduces the term highly secure software. Subsequently, it describes a large part of the reviewed and analyzed guidelines, recommendations, approaches and techniques for data protection during storage, management and transfer between mobile devices that are useful to support the design and implementation of a HSS, including to support the definition of concrete HSS security evaluation criteria.

2.2.1. Introduction

The term software security can be defined informally as an attribute describing the ability of this software to protect itself from external attacks (accidental or intentional) [14]

The term highly secure software (HSS) can be defined as follows – a security-reliable software that is:

- able to fully satisfy the Fundamental Security Requirement for Data Protection (ОИСЗД);
- fully capable to adequately protect itself and the data it is operating with, including to continue to work correctly according to its intended purpose by design, the stakeholders needs and without compromising the security of the data trusted to it, even in an event of an error, an incident, an accident, a malicious behavior and/or deliberate attacks (all together or separately);
- have a much higher level of data security compared to a regular general-purpose software;

Essential to the design and implementation of HSS is to be ensured that all architectural decisions are carefully and reasonably taken. In addition this, it is crucial to be ensured a correct usage of proven approaches, methods, methodologies, cryptographic tools and techniques in accordance with the specifics of the execution environment, the lifecycle of the data and the potential data security threats (depending on software's application area) at an architectural level and without a violation of any known constraints and limitations. In the next sections is provided an overview of key data protections approaches, standards, tools and techniques, and also concrete guidelines and recommendations on their correct usage.

2.2.2. Cryptographic Tools and Techniques for Local Data Protection

The symmetric-key cryptography is the right choice *w*hen in the context of HSS it is needed to be ensured a strong local data protection in order to be possible for the data to be securely long-term stored on the device on which the concrete HSS instance is running.

There are algorithms whose security requires to keep secret both the essence of the algorithm and its principle of operation [15]. Such algorithms are called restricted algorithms and their usage as a primary tool for providing security is considered as totally unreliable and inadequate [15]. Moreover, given the significant progress around the reverse-engineering tools, and especially given that the issues related to the restricted algorithms are not only well known, but also solved by the modern cryptography [15], the following conclusion can be drawn: it is strictly prohibited for the security core of a given HSS to be build on top of security mechanisms, modules, subsystems, tools and techniques relying on restricted algorithms as a main method of providing security.

The modern ciphers using a symmetric key are generally divided into: stream ciphers and block ciphers in a certain mode of operation.

Stream ciphers are further divided into: synchronous stream ciphers and self-synchronizing stream ciphers [16]. As part of the dissertation work several significant documents used for stream

ciphers standardization are reviewed. In addition, attention is paid to the fact that the standardized ciphers reviewed are able to provide in their base form a confidentiality only, but unable to provide a reliable protection against data tampering, for which it is needed for these cipher to be combined with an additional cryptographic primitive known as a message authentication code (MAC). Also, several well-known MAC implementation are reviewed, including the particularly well-known combination of *ChaCha20* with *Poly1305*, suitable for local data protection in the context of HSS given that their correct usage is ensured.

With regard to block ciphers, it is noted that, in general, a block cipher by itself is able to describe a cryptographic transformation suitable for a one-time encryption of only one data block with a single fixed key. The need to apply the block cipher in an appropriate mode of operation is indicated, including the need to select an appropriate cipher in a given mode of operation in order to be possible to be ensured that the desired security goals are achievable. Some of the most popular block ciphers implementations and modes of operation are reviewed. Existing publicly available cryptanalysises, including vulnerabilities databases, are reviewed to screen out some well-known implementations of block ciphers / modes of operation that are considered insecure for modern applications (such as the following ciphers: *Data Encryption Standard* (*DES*), *Triple Data Encryption Algorithm* (*TDEA*), *Blowfish* and modes of operation: *Electronic Codebook* (*ECB*)). Also, some well-known block ciphers / modes of operation especially suitable for local data protection in the context of HSS are reviewed (such as the following ciphers: *AES* (*Advanced Encryption Standard*), *Twofish*, *Camellia* and modes of operation: *Galois/Counter Mode* (*GCM*)).

2.2.3. Cryptographic Tools and Techniques for Secure Export, Transfer and Import of Data

The public-key cryptography combined together with the symmetric-key cryptography is the right choice *w*hen in the context of HSS it is needed to be ensured data protection for data that are subject to secure export, transfer and import. Given that, this section provides an overview of existing public cryptanalysises, including vulnerabilities reports about well-known public-key cryptographic algorithms implementations. In addition to this, some proven guidelines and recommendations on the topic are reviewed.

In regard to known insecure implementations of existing public-key cryptographic algorithms, the dissertation provides an overview of several historically interesting algorithms, paying particular attention to their concrete weaknesses that render them totally inadequate for modern applications in the context of HSS.

In regard to the public-key cryptographic algorithms that can be considered both secure and applicable to the real world, the dissertation draws attention to the fact that one part of these algorithms is only suitable for key exchange, another part is suitable for data encryption / decryption (and possibly for key exchange by an extension), and a third part is only suitable for digital signatures. Also, it is noted that there a relatively small number of algorithms that are suitable for both encryption / decryption and digital signatures activities. In addition, one of the most famous public-key algorithms, namely *RSA* (*Rivest–Shamir–Adleman*), suitable for both encryption / decryption and digital signatures activities is reviewed. Related to this, some notable studies, including cryptanalysis about *RSA*, are also analyzed, and concrete security weaknesses are identified. Based on the analysis, the following conclusion is drawn: for *RSA*, to be considered secure and reliable enough for modern applications, it is critical to be chosen a secure implementation that is addressing all known weaknesses and limitations of *RSA* in order to be possible an already known issues prevention.

Related to the implementation of secure data export, transfer and import functionality in the context of a HSS, it is crucial to be noted that choosing a secure public-key algorithm alone cannot be considered sufficient in itself. That means, even when the algorithm chosen is secure enough, it should be properly used by taking into consideration the specifics of the data and the transportation medium, and without allow for a violation of any known limitations and constraints related to them. A common approach to implement secure data export, transfer and import functionality in the context of a HSS is to combine one or more public-key cryptographic algorithms with one or more symmetric-key cryptography algorithms.

2.2.4. Use of Cryptographic Tools and Techniques, Combined Together with Programming Approaches and Principles, including Secure Hardware Protections for Ensuring Full Data Protection

Regardless of the specific data type, for which a given HSS is expected to operate with, it should be able to ensure its secure operation by meeting a number of security requirements, not least of which is the Fundamental Security Requirement for Data Protection (ОИСЗД). This, in turn, requires a number of considerations to be made in order to be enforced at architectural level that the whole functionality related to HSS's operation with universal data of an arbitrary type will actually be designed and implemented securely with a strong data protection ensured and also with the HSS being able to take into consideration the concrete data type in use, the specifics of the data and their lifecycle. At the core of these consideration is the correct choice of proven cryptographic algorithms, techniques, approaches, standards and tools for data protection. With this in mind, as part of dissertation work were reviewed a number of documents that can serve as a good starting point on the topics related, especially in the context of HSS.

Although the correct choice of proven cryptographically strong algorithms, tools and techniques is particularly important security consideration, the dissertation notes that this consideration alone is not sufficient to satisfy the Fundamental Security Requirement for Data Protection (ОИСЗД), especially in the context of HSS. More specifically, a special attention is paid to be noted that it is needed to be built a complete cryptosystem in the context of HSS in order to be possible for the Fundamental Security Requirement for Data Protection (ОИСЗД) to be met. In addition to this, it is noted that the cryptosystem by itself should be able to ensure that all cryptographic tools, techniques, approaches, principles and design patterns combined together will always be used correctly and without a violation of any known their limitations. In other words, that means no violations of rules about cryptographic keys correct use / reuse, as well as the different security aspects like cryptographic key cryptoperiods, key management / key size activities should be possible, including those described in document [17], section 3.6, 5.2, 5.3, 6.1 и 6.3 of document [18], as well as the documents [19], [20], [21] and [22]. In addition, fundamental to the construction of a reliable cryptosystem in the context of HSS are also the guidelines and recommendations for the secure storage and operation of cryptographic keys that should be viewed as important as those related to secure long-term storage of trusted data. A widely accepted and proven approach for ensuring strong cryptographic keys protection is this based on the concept about secure cryptographic keys storage and operation by enforcing the use of a secure container known as a *keystore*. This approach can be implemented on an application level basis by implementing a separate module or subsystem as part of the HSS's cryptosystem itself. Also, depending on the technical capabilities of the device and its operating system, it is possible for some of the security critical operations related to secure storage and operation of cryptographic keys to be delegated to a completely independent separate system that is able to take advantage of the capabilities of certain secure hardware running in complete isolation from both the HSS itself and the device's underlying operating system. The main advantage of a keystore relying on such type of partial delegation is that the HSS is actually able to take advantage of the capabilities of a specialized secure hardware such as Trusted Execution Environment (TEE), Secure Element (SE) and many others. However, secure hardware is not always available and even when available its capabilities might be limited, for example the Android APIs in regard to StrongBox are limiting the RSA key size up to 2048 bits [23].

A well-known approach to implement strong cryptographic keys protection is the security credentials based key derivation approach, where a base symmetric key is derived from the security credentials in order to be possible for this key to be used together with a strong cipher for protecting the other cryptographic keys and also to make possible their secure long-term storage. The cryptographic key derived is typically called Main cryptographic key (MCK). This key is responsible to protect the other cryptographic keys. The method used for its derivation is known as a method based on a key derivation function (*KDF*). There a lot of options for choosing a KDF with the following important consideration to be noted: the choice of a proven well-known KDF, together with its correct usage (without violation of known limitation of the KDF chosen) and its correct configuration ensuring appropriately set parameters depending on the desired security level, are crucial for providing strong data security. In the dissertation work an attention is paid on a KDF that is derived from one of the most common types of security credentials, namely a master password. More specifically, as part of the

dissertation work are reviewed significant documents with recommendations and guidelines on correct usage of *Password-based Key Derivation Function 2* (*PBKDF2*) and *Argon2*.

Last but not least, it is essential to consider the benefit of implementing a strategy to minimize the consequences if some cryptographic key is being compromised. With this in mind, concepts such as Data Protection Key (DPK) and Secure hardware-based data protection key (SHBDPK) are introduced as part of the dissertation work. The dissertation also bolds the need for a HSS to be built upon a modular software architecture with a clear separation of concerns, capable to ensure the correct usage of appropriate programming approaches, together with significant principles, techniques and security strategies in order to be improved not only the security and the reliability of the HSS that is being developed, but also to be improved its maintainability and extensibility, but without compromising the security.

2.2.5. Processes for Reviewing Existing Cryptographic Tools, Techniques and Standards

There is a plenty of proven cryptographic tools, techniques and standards available. Despite this, it is particularly important to be ensured that these tools, techniques and standards are regularly reviewed over certain periods of time in order to be possible for them to be timely audited by taking into consideration any new information about their security, including things like emerging cryptanalysis, technology development rate, potential for weaknesses identification, etc. As part of the dissertation work, it is provided an example of a standardized process for reviewing existing cryptographic tools, techniques and standards. Related to this, the Crypto Publication Review Project by the U.S. National Institute of Standards and Technology (NIST) is reviewed. In addition, the dissertation notes that public initiatives related to quantum-resistant cryptography are still work in progress, citing algorithms listed by the U.S. NIST. Moreover, it concludes that it is very important for each HSS to be ensured that it will be designed and implemented upon a strong security architecture allowing potential transition to upgraded versions of the cryptographic tools, techniques and standards on which it relies with an option for a potential migration to future ones, like these provided by the Post-Quantum Cryptography.

2.3. Overview of Reverse-Engineering Tools and Frameworks Used

This section describes some of the tools (*Bytecode Viewer*, *IDA Pro*, *Frida*, *Volatility*) that have been studied for the needs of the experiments and digital investigation planed as part of the dissertation work, especially helpful to support a potential security analysis process and also the activities related to it (like potential vulnerabilities identification and reverse-engineering), useful when it is needed be revealed some design / implementation secrets or any other information for a given software (including closed-source software) that is hidden or secret, but can be very helpful, especially in case that it is needed to be: confirmed the behavior of the software examined, including to be compared its actual behavior to the its behavior officially documented in the official sources like documentations, technical papers, etc; revealed some significant details about the design and implementation of the specific mechanisms / modules / systems and subsystems of the examined software with the goal to support a potential security analysis / evaluation process.

2.4. Conclusion and Key Findings

In this chapter, an overview is made, on the basis of which an analysis of the current state in the researched field is carried out. Special attention has been paid to issues related to data security, security requirements and data security threats, and a number of scientific works and books in the field have been reviewed and analyzed. Main data security threats in the context of HSS are identified and subsequently defined. In addition to this, some basic concepts, terms and definitions on the topics about data security are introduced with the goal to support the future research planned, included the experiments planned. A fundamental security requirement for data protection (ОИСЗД) in the context of HSS is derived and specified. Also, a number of recommendations, guidelines, approaches, principles, techniques and standards for data protection during storage, management and transfer between mobile devices are examined. Based on this, an analysis was carried out and the cryptographic tools and techniques in the context of HSS were classified in two groups: “Cryptographic Tools and

Techniques for Local Data Protection" and "Cryptographic Tools and Techniques for Secure Export, Transfer and Import of Data". Known processes for revising existing cryptographic tools, techniques and standards were reviewed and analyzed, including projected expectations for future developments in the field and the associated implications in regard to data protection in the context of HSS. Last but not least, this chapter presented some of the tools studied as part of the dissertation work with the goal to be supported the digital investigations planned, including the related to them experiments and activities planned.

In conclusion, the key findings for this chapter can be summarized as follows:

1. The Fundamental Security Requirement for Data Protection (ОИСЗД) defined in this chapter sets the lower bound for security in regard to an HSS, i.e. it sets the absolute minimum in terms of security. However, it is important to be noted that depending on the specific type of HSS being developed and the needs of the stakeholders, this requirement should be supplemented;
2. It is strictly prohibited for the security core of a given HSS to be build on top of security mechanisms, modules, subsystems, tools and techniques relying on restricted algorithms as a main method of providing security;
3. It is very important for each HSS to be ensured that it will be designed and implemented upon a strong security architecture allowing potential transition to upgraded versions of the cryptographic tools, techniques and standards on which it relies with an option for a potential migration to future ones, like these provided by the Post-Quantum Cryptography;
4. Based on the research and analysis carried out in this chapter, it is essential to be defined specific HSS security evaluation criteria allowing to be check to what extent the currently existing HSS is taking into consideration the well-known and proven recommendations, guidelines, approaches, principles, techniques and standards for data protection;

Chapter 3: Study, Comparative Security Analysis and in Breadth Security Evaluation of Highly Secure Software

In this chapter, a methodology is proposed for conducting an investigation, comparative security analysis and in breadth security evaluation of HSS at high-level. Subsequently, it describes the high-level study of HSS made as part of the dissertation work, together with the comparative security analysis and in breadth security evaluation of a concrete PMS / DVS made on the basis of concrete security criteria defined.

3.1. Defining a Methodology for Conducting Investigation, Comparative Security Analysis and in Breadth Security Evaluation of Highly Secure Software at a High-Level

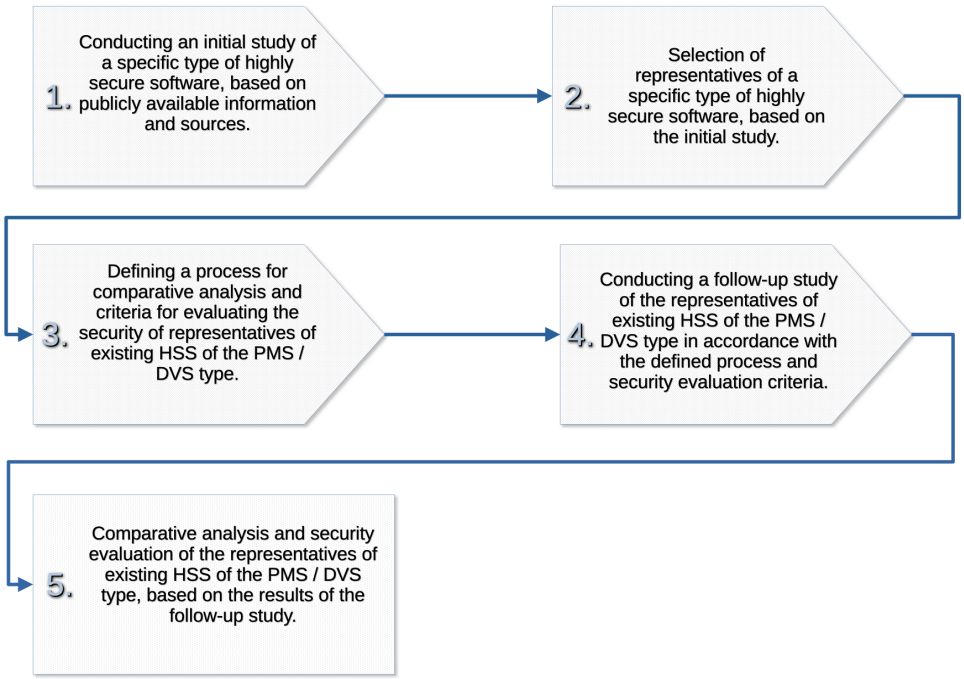

Figure 2: Methodology for Conducting Investigation, Comparative Security Analysis and in Breadth Security Evaluation of HSS at a High-Level

Figure 2 graphically presents a methodology for Conducting Investigation, Comparative Security Analysis and in Breadth Security Evaluation of HSS at a High-Level. Its main purpose is to support the comprehensive study of a specific type of HSS for conducting a comparative security analysis with a security assessment. The focus is on evaluating the software's security from the perspective of assessing to what extent the studied software is capable of protecting its entrusted data based on specific security evaluation criteria. The methodology includes the implementation of multiple steps, grouped in the five main phases and shown on Figure 2.

3.2. Study, Comparative Security Analysis and in Breadth Security Evaluation of Password Management Software / Digital Vault Software

This section applies the methodology defined in Section 3.1 in order to be conducted an investigation, together with a comparative security analysis and in breadth security evaluation of PMS / DVS.

3.2.1. Initial Study and Representative Password Management Software / Digital Vault Software Selection

After an initial high-level research, it was found out that part of the Android applications examined have already been studied / analyzed by independent security researchers. Based on an

analysis of the results of these researchers, it was found that some of the applications are relying on security through obscurity as a main method of providing security or weak encryption / hashing techniques. An example for such applications are: *AppLock* (*package name: com.domobile.applock*); *Keepsafe* (*package name: com.kii.safe*); *Keepsafe* (*package name: com.kii.safe*); *Calculator Vault* (*package name: com.calculator.vault*);

Continuing the high-level study of popular Android applications identifying themselves as PMS / DVS, such applications were also found for which no evidence was found indicating that these applications rely on security through obscurity as a main method of providing security and /or other similar techniques / approaches. From these, in accordance with the prescriptions of the methodology defined in Section 3.1, a representative sample of the following 4 applications was selected for further analysis and security evaluation: *Keeper* (*com.callpod.android_apps.keeper*); *LastPass* (*com.lastpass.lpandroid*); *Dashlane* (*com.dashlane*); *Bitwarden* (*com.x8bit.bitwarden*);

3.2.2. Defining a Process for Comparative Security Analysis and Security Evaluation of Password Management Software / Digital Vault Software

In order to help define the essential process of comparative security analysis and security evaluation, there are three sets of security evaluation criteria (MKOC) defined as part of the dissertation work. These sets are presented on Figure 3 below .

MKOC_№1 : Security measures for attack surface reduction – This set of SEC checks whether the examined application is applying efficient security measures for attack surface reduction in order to prevent potential security failures. It consists of the following 10 criteria:

KOC_№1.1 : Configured by default to work completely offline;

KOC_№1.2 : Configured by default to prevent screen capturing;

KOC_№1.3 : Configured by default to exclude views containing trusted data from Android assistant, accessibility services, auto backup and the other similar to them services;

KOC_№1.4 Configured by default to use its own input method editor (IME) implementation that is security enhanced;

KOC_№1.5 : Security measures to prevent security credentials leakage;

KOC_№1.6 : Security measures to prevent MCK leakage;

KOC_№1.7 : Strategy to prevent potential trusted data leakage into the shared clipboard;

KOC_№1.8 : Strategy to prevent potential trusted data and cryptographic keys leakage from main memory;

KOC_№1.9 : Trusted data protection strategy based on protection with multiple specialized keys of type DPK;

KOC_№1.10 : Strategy for full cryptographic keys rotation when security credentials are changed;

MKOC_№2 : **Security measures for protection of the trusted data long-term stored in the persistent local storage** – This set of SEC checks whether the examined application is applying efficient security measures to protect its most valuable asset while held in the persistent local storage. It consists of the following 7 criteria:

KOC_№2.1 : Enforces hardware-backed protection of the trusted data by requiring additional cryptographic transformations to the trusted data inside a secure hardware like TEE when such hardware is available;

KOC_№2.2 : Cryptographic algorithm used to ensure the confidentiality of the raw values (CC);

KOC_№2.3 : Cryptographic algorithm used to ensure both the authenticity and the integrity of the raw values (CC);

KOC_№2.4 : In addition to the standard cryptographic keys of type DPK, the software is able to protect the raw values (CC) with one or more SHBDPK;

KOC_№2.5 : Cryptographic algorithm used to ensure the confidentiality of the Metadata;

KOC_№2.6 : Cryptographic algorithm used to ensure both the authenticity and the integrity of the Metadata;

KOC_№2.7 : SHBDPK are used to protect the Metadata;

MKOC_№3 : **Security measures for protection of the cryptographic keys** – This set of security evaluation criteria checks whether the examined application is applying efficient security measures to protect the cryptographic keys used to protect the trusted data. It consists of the following 5 criteria:

KOC_№3.1 : Enforces security credentials minimum length in order to be ensured a larger amount of entropy;

KOC_№3.2 : KDF used to generate the MCK;

KOC_№3.3 : Configured by default to enforce cryptographic keys exhaustion control;

KOC_№3.4 : Cryptographic algorithm used to ensure the confidentiality of the cryptographic keys stored in the persistent local storage;

KOC_№3.5 : Cryptographic algorithm used to ensure both the authenticity and the integrity of the cryptographic keys stored in the persistent local storage;

Figure 3: Sets of Security Evaluation Criteria for Password Management Software / Digital Vault Software.

The essential process of comparative security analysis and security evaluation is defined with the help of the rules provided as part of the dissertation work. In general, two types of values are used to assign a specific score for each criterion / set of criteria: Quantitative Value (QV) and String Value (SV).

3.2.3. Security Evaluation, Findings and Results Based on the Study and Comparative Security Analysis Conducted for the Representative Sample of Password Management Software / Digital Vault Software

This section presents the comparative security analysis and evaluation results for the representative sample from Section 3.2.1. The goal here is to security analyze and security compare the software from the representative sample based on the results. Subsequently a security evaluation report is being prepared on the basis of which a decision is taken on how to proceed with the next studies and experiments planned.

Identifier of		*Keeper* [24], [25]		*LastPass* [26]		*Dashlane* [27]		*Bitwarden* [28]–[31]	
Set	**Criterion**	**SV**	**QV**	**SV**	**QV**	**SV**	**QV**	**SV**	**QV**
MKOC_№1	**All together**	Not satisfied	10	Not satisfied	10	Not satisfied	10	Partially satisfied	20
	KOC_№1.1	No	0	No	0	No	0	No	0
	KOC_№1.2	Yes	1	Yes	1	Yes	1	Yes	1
	KOC_№1.3	No	0	No	0	No	0	No	0
	KOC_№1.4	No	0	No	0	No	0	No	0
	KOC_№1.5	No	0	No	0	No	0	No	0
	KOC_№1.6	No	0	No	0	No	0	No	0
	KOC_№1.7	No	0	No	0	No	0	No	0
	KOC_№1.8	Unknown	0	Unknown	0	No	0	Yes	1

Identifier of		*Keeper* [24], [25]		*LastPass* [26]		*Dashlane* [27]		*Bitwarden* [28]–[31]	
Set	**Criterion**	**SV**	**QV**	**SV**	**QV**	**SV**	**QV**	**SV**	**QV**
	KOC_№1.9	No	0	No	0	No	0	No	0
	KOC_№1.10	Unknown	0	Unknown	0	Unknown	0	No	0
MKOC_№2	**All together**	Partially satisfied	28,57	Not satisfied	14,29	Partially satisfied	28,57	Partially satisfied	28,57
	KOC_№2.1	Unknown	0	Unknown	0	Unknown	0	No	0
	KOC_№2.2	AES/GCM, 256-bit key	1	*AES/CBC*, 256-bit key	1	*AES/CBC*, 256-bit key	1	*AES/CBC*, 256-bit key	1
	KOC_№2.3	*AES/GCM*, 256-bit key	1	No	0	*HMAC*	1	*HMAC-SHA256*, 256-bit key	1
	KOC_№2.4	Unknown	0	Unknown	0	Unknown	0	No	0
	KOC_№2.5	Unknown	0	Unknown	0	Unknown	0	No	0
	KOC_№2.6	Unknown	0	Unknown	0	Unknown	0	No	0
	KOC_№2.7	Unknown	0	Unknown	0	Unknown	0	No	0
MKOC_№3	**All together**	Partially satisfied	40	Partially satisfied	40	Partially satisfied	40	Partially satisfied	80
	KOC_№3.1	Yes	1	Yes	1	Yes	1	Yes	1
	KOC_№3.2	*PBKDF2* (PRF = *SHA-512*, iterations = 100 000)	1	*PBKDF2* (PRF = *SHA-256*, iterations = 100 100)	1	*Argon2d* (parallelism = 2, memory = 32MB, iterations = 3)	1	*PBKDF2* (PRF = *SHA-256*, iterations = 100 000)	1
	KOC_№3.3	Unknown	0	Unknown	0	Unknown	0	No	0
	KOC_№3.4	Unknown	0	Unknown	0	Unknown	0	*AES/CBC*, 256-bit key	1
	KOC_№3.3	Unknown	0	Unknown	0	Unknown	0	*HMAC-SHA256*, 256-bit key	1
All together	**All together**	Partially satisfied	26,19	Partially satisfied	21,43	Partially satisfied	26,19	Partially satisfied	42,86

Table 1: Comparative Security Analysis and Evaluation Results for the Representative Sample Software from Section 3.2.1

Table 1 summarizes the results of the comparative security analysis and evaluation for the software from the representative sample selected in Section 3.2.1.

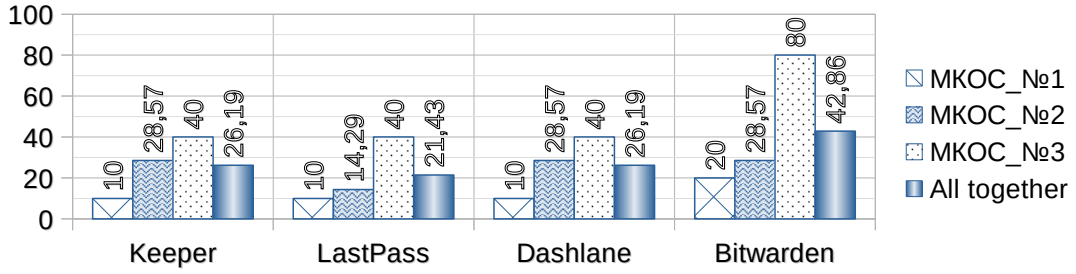

Figure 4: An average satisfaction rate of each criteria set

A graphical representation of the quantitative results from the comparative security analysis and evaluation that were performed can be seen on Figure 4. According to the results, *Bitwarden* appears to be the most secure PMS / DVS among the examined ones, *LastPass* appears to be the least secure and *Keeper* and *Dashlane* are given equal scores according to the criteria.

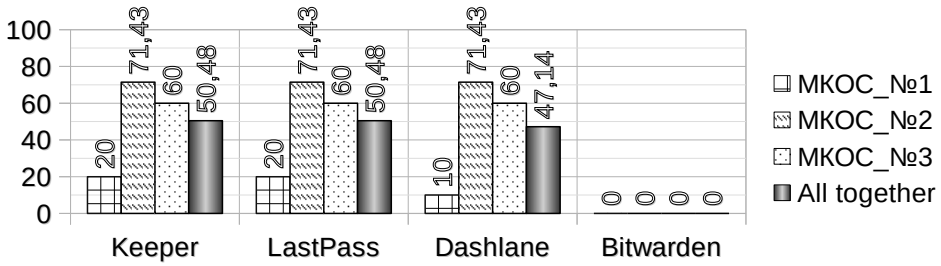

*Figure 5: An average rate of **Unknown** values for each criteria set*

Figure 5 is showing the average percentage of **Unknown** values given to each criteria set and the average percentage of **Unknown** values given to all criteria together. Being an open source software, *Bitwarden* is providing a better transparency and trustworthiness compared to the other applications examined. Among the closed source software applications, *Dashlane* appears to be the most well documented application, followed by *Keeper* and *LastPass*, which are given equal scores according to the result.

3.3. Conclusion and Key Findings

This chapter defined a methodology for conducting an investigation, comparative security analysis and in breadth security evaluation of HSS at a high-level. Subsequently, the applicability of the methodology defined was validated by applying it to study, security compare and in breadth security evaluate a representative sample of PMS / DVS. More specifically, a high-level analysis of publicly available official information, including technical documentation, results of independent security researchers, and other publicly available sources, was performed for a number of known *Android* applications of type PMS / DVS. Based on the analysis, a representative sample of 4 software applications of type PMS / DVS was selected, after which a comparative security analysis and security evaluation process was defined, together with specific security evaluation criteria. In accordance with the process and criteria defined, the necessary research and experiments were carried out. Subsequently, it is prepared a high-level comparative security analysis and security evaluation report for the applications from the representative sample, containing also an information about the studied applications security awareness and transparency degree. According to the results, *Bitwarden* appears to be the most secure PMS / DVS among the examined ones and also the application having the best transparency and trustworthiness. However, *Bitwarden* failed to fully satisfy all criteria sets.

In conclusion, the key findings for this chapter can be summarized as follows:

1. None of the software applications studied managed to fully satisfy all sets of security evaluation criteria;
2. There are a number of important technical details about core password management / vault software concepts and functionality missed from the official documentations of all examined closed source applications and this:
   - **2.1.** Is a serious problem for the transparency and trustworthiness themselves, especially if the end-user have the technical competency and proficiency to make an informed and reasoned choice before possibly purchasing given PMS / DVS;
   - **2.2.** Casts doubts on how much the end user can trust them;
3. It is needed, a future in-depth security investigation, during which with the help of specialized reverse-engineering tools and techniques the runtime behavior of the examined applications, their principles of operation as well as their actual implementations should be carefully analyzed;

Given the conclusions drawn, further research in the dissertation is planned to deepen and focus on one of the most key aspects of security for PMS / DVS, namely the aspect about both protecting the data in RAM areas belonging to the examined software and limiting the sensitive data exposure in these areas. Also, in regard to the further research planned, it was decided for the representative sample size to be reduced to two software applications only – these with the highest security rating (based on the results in Section 3.2.3).

Chapter 4: An Investigation on In-Memory Data Protection Effectiveness in the Context of Highly Secure Software Execution

This chapter describes the digital investigation conducted as part of the research and the experiments made, including the comparative security analysis and security evaluation performed (in regard to the efficiency of the protection of data entrusted to the examined software and stored in its main memory areas.)

4.1. Define a Main Memory Inspection Methodology With Regard to the Efficiency of In-Memory Data Protection in the Context of HSS Execution

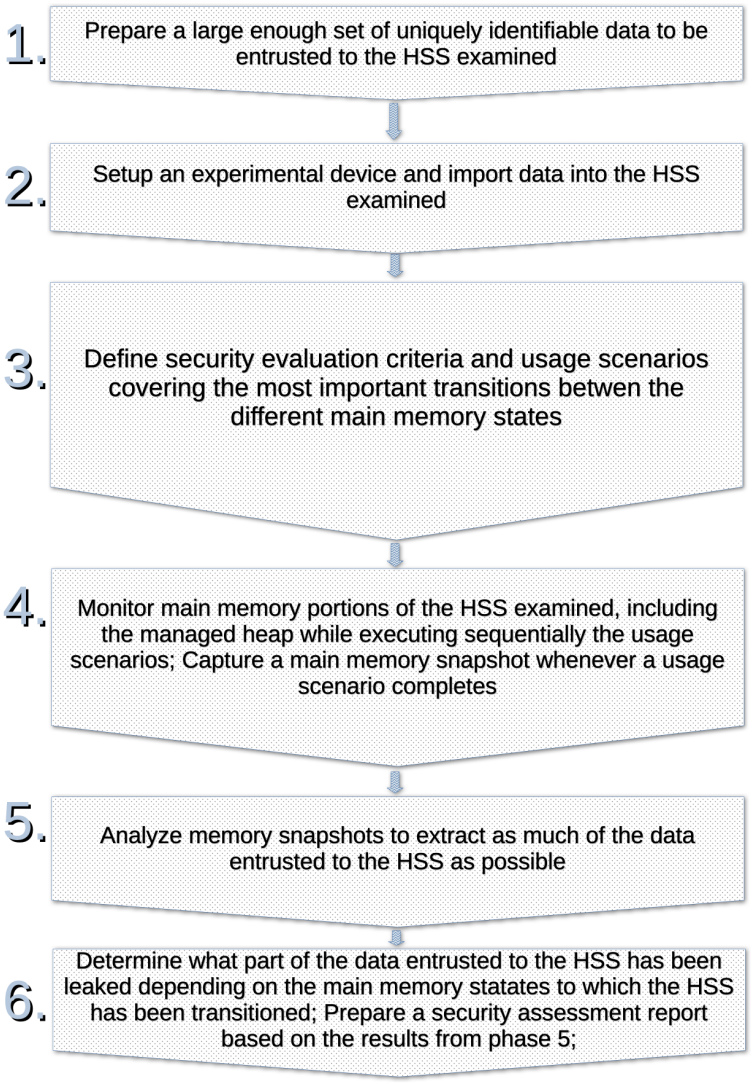

Figure 6: Main Memory Inspection Methodology With Regard to the Efficiency of In-Memory Data Protection in the Context of HSS Execution

The overview made as part of Section 2.1.4 have confirmed the actuality of the need for application level supplementary measures for prevention of sensitive data exposure in main memory by having applied timely use of both secure in-memory data encryption and secure data deletion (whose implementations are able to ensure that the concrete HSS instance will be capable to take an advantage of device's secure hardware when that is both possible and reasonable). Related to this, and in order to check to what extent the existing HSS manages to protect its data while held in RAM, this section defines a methodology for main memory inspection and in-memory data protection effectiveness measurement in the context of HSS execution, applicable for evaluating the effectiveness of protection of data entrusted to a concrete HSS, and held in main memory areas belonging to the process in context of which the HSS examined is running. The methodology itself is graphically presented on Figure 6. As can be seen on the figure, there are six main phases of the methodology, and each successive phase uses the results of the previous phase. In the dissertation, the prescriptions of each of the phases are described, and in addition to this, additional definitions and concepts are introduced, such as:

- **Main memory states (СОП) definitions:**
  СОП1_УНИЩОЖЕН;
  СОП2_НЕДОВРЕНО_ИЗПЪЛНЕНИЕ;
  СОП3_ДОВЕРЕНО_ИЗПЪЛНЕНИЕ;
- **Set of main memory states (МСОП)**:

МСОП = {СОП1_УНИЩОЖЕН, СОП2_НЕДОВРЕНО_ИЗПЪЛНЕНИЕ, СОП3_ДОВЕРЕНО_ИЗПЪЛНЕНИЕ}

- **Main memory states transition (ПСОП);**
- **Sequence of Main Memory States Transitions (ППСОП);**

On Figure 7 is shown a template for describing a specialized usage scenario depending on the specifics of a given software and data entrusted to it. This template is used to support the definition of specialized usage scenarios in regard to phase 3 of methodology shown on Figure 6.

${идентификатор_на_сценарий} : ${заглавие_на_сценарий}

Expected Sequence of Main Memory States Transitions:

1. ${преход$_1$}
…
n.${преход$_n$}

Prerequisite::

${идентификатор_на_условие$_1$}; … ${идентификатор_на_условие$_u$};

Steps:

1. ${стъпка$_1$}
…
m.${стъпка$_m$}

Figure 7: Template for Describing a Specialized Usage Scenario Depending on the Specifics of a Given Software and Data Entrusted to It

4.2. The Design and Implementation of a Software Tool for Inspecting Main Memory Snapshots of Software's Private Areas in Main Memory

4.2.1. Conceptual Model and Technologies Used

In this section, the specialized software tool that is developed as part of the dissertation work is presented. The main goal of the tool is to support the automation of identification and extraction process in regard to identification and extraction of trusted data exposed in the RAM, based on snapshots of the RAM. Conceptually, the tool is developed with the idea of universal applicability, i.e. to allow for inspection of snapshots of the RAM areas of a software of user choice depending on JSON based configurations, allowing to set up the tool according to the specifics of the examined software. In addition, the tool provides support for creating logical groups for both the lookup data and RAM areas snapshots, with an option to be described their mapping to predefined usage scenarios. The main technologies used to develop the tool are: *JVM*, *Kotlin* и *Gradle*. For tool's declarative configurations is used *JSON* and *CSV is* used for describing the structure of lookup data and for loading its raw values.

4.2.2. Functional Requirements

As a first step in the design and implementation process, it was taken a decision to specify the functional requirements for the tool. These requirements are summarized bellow:

- The tool should provide a functionality for universal inspection of the textual content of RAM areas snapshots for a running software instance of a user choice;
- The tool should provide a support for inspection report preparation functionality, including to be able to assist the analysis process in regard to the inspection performed; also the tool should be able to both identify and extract all trusted data contained into a memory snapshot; in addition, the report prepared should have the full information about the total number of values found for each field based on the structure of the trusted data, the exact values that were found, and the exact identifiers of the RAM snapshots from which these values were retrieved, including the identifiers of the respective usage scenarios mapped to the snapshots taken;
- The tool should allow the user to declaratively configure the functionalities related to the universal inspection;

- The tool functionalities must comply with the prescriptions of the methodology defined in Section 4.1;

4.2.3. Non-functional Requirements

Among the non-functional requirements specified as part of the dissertation work are:

- The tool should be implemented on the basis of high-performance and efficient algorithms, allowing the search and inspection of multiple RAM snapshots for a large number of values from trusted data records (at least 50 000 string values of at least 10 characters each, grouped into at least 10 000 records, each having at least 5 fields);
- The tool should be built upon a software architecture enforcing modular design with a clear separation of concerns; the APIs provided should be simple and concise; maintenance and expansion in terms of both modifying existing functionality and adding a new functionality should be possible;
- The tools quality attributes must comply with the prescriptions of the methodology defined in Section 4.1;

4.2.4. General Architecture, Design and Implementation Notes, and End-User Instructions.

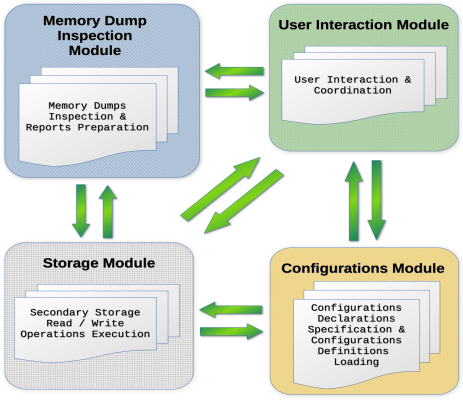

Figure 8: Software Tool Architecture

`

The general architecture of the software tool that was designed and developed as part of the dissertation work is shown on Figure 8. The tool has 4 main modules:

- ***User Interaction Module*** : This is the tool's entry point. It is responsible to manage the end-user interaction process. More specifically, it serves as a communication layer that is responsible to coordinate the communication between the tool's modules and the end user.
- ***Configurations Module*** : This module is responsible to manage the activities that are related to the configuration declarations and configuration definitions specification process. It relies on the *Storage Module* for filesystem related activities, for example, to read / write certain configuration declarations / definitions from / in filesystem.
- ***Storage Module*** : This module is responsible to perform the necessary input-output operations for reading data from / writing data in filesystem. It also manages the serialization / deserialization related activities, for example, this module is responsible to deserialize the configuration definitions written in JSON.
- ***Memory Dump Inspection Module*** : This module is responsible to perform the actual functionality of search and inspection for arbitrary snapshots of the RAM areas belonging to the investigated software.

```
java -jar MemoryDumpInspectionTool.jar ${път_до_конфигурационен_файл}
```

Figure 9: The Command Line Interface of the Tool

The tool's command line interface (CLI) is shown on Figure 9. In its simplest form, the tool can takes one argument, that is: **${път_до_конфигурационен_файл}** – the full path to a certain configuration definitions file.

4.3. Digital Investigation, Security Analysis and Evaluation

4.3.1. Applications Selection

On the basis of the comparative security analysis and the security assessment carried out in Chapter 3, and in accordance with the conclusions made in Section 3.3, in the dissertation it was decided to conduct a digital investigation for the two software applications with the highest security scores (based on the results in Section 3.2.3), namely the software applications: *Bitwarden* and *Keeper*.

4.3.2. Device Setup and Data Preparation

In accordance with phase 1 and phase 2 of the methodology (shown on Figure 7 in Section 4.1) and as part of the preparation for the digital investigation, the following actions were performed:

1. A new Android virtual device emulating Pixel 3 device was created.
2. The latest versions of the applications from representative sample weredownloaded from Google Play Store and then installed on the emulatedtest device.
3. Two free accounts were created.
4. A large set of data to be entrusted to the applications from representativesample was prepared and imported. It consists of the following a *CSV* file containing 10 000 text onlyrecords representing synthetic login data for external sites (the data was generated by a special tool written in Kotlin).
5. Two records containing file attachments were created in *Keeper*.

4.3.3. Defining Security Evaluation Criteria with Regard to the Efficiency of In-Memory Data Protection in Highly Secure Software

This section defines a set of security evaluation criteria with regard to the efficiency of in-memory data protection in highly secure software (МКОЕЗДВССОП). The criteria defined can be summarized as follows:

МКОЕЗДВССОП_1 : Security measures for limiting exposure of ДД in main memory – This set of criteria checks whether the software under investigation implements effective security measures to minimize the amount of ДД exposed in cleartext in its owned areas in the main memory. This set is made of the following two sets:

МКОЕЗДВССОП_1.1 : Security measures for limiting exposure of highly confidential strings in main memory – It checks whether the software under investigation undertakes adequate security measures to protect the confidentiality of highly confidential strings in its owned areas in the main memory, by taking immediate actions to minimize their exposure in cleartext there, including their complete deletion immediately after use. This set comprises 5 criteria, described in brief below:

КОЕЗДВССОП_1.1.1 : HSS is allowed to load a highly confidential string in cleartext in main memory only if strictly necessary, with immediate action taken for secure deletion of the string as soon as the strict necessity ceases;

КОЕЗДВССОП_1.1.2 : Targets the security measures in the specific case where the instance of the software under investigation continues to run after a procedure of standard user-requested logout has been completed;

КОЕЗДВССОП_1.1.3 : Targets the security measures in the specific case where the user performs a FORCE STOP;

КОЕЗДВССОП_1.1.4 : Targets the security measures in the specific case where the instance of the software under investigation continues to run after a procedure of implicit logout / implicit vault lockout, triggered by the built-in auto-lock protection due to inactivity, has been completed;

КОЕЗДВССОП_1.1.5 : Targets the security measures in the specific case where the user removes the instance of the software under investigation from the system's screen of recently run applications immediately after a procedure of implicit logout / vault lockout, activated by the built-in auto-lock protection due to inactivity, has been completed;

МКОЕЗДВССОП_1.2 : Security measures for limiting exposure of confidential strings in main memory – It checks whether the software under investigation undertakes adequate security measures to protect the confidentiality of confidential strings in its owned areas in the main memory, by taking immediate actions to minimize their exposure in cleartext there, including their complete deletion immediately after use. This set comprises 5 criteria, described in brief below:

КОЕЗДВССОП_1.2.1 : HSS is allowed to load a highly confidential string in cleartext in main memory only if strictly necessary, with immediate action taken for secure deletion of the string as soon as the strict necessity ceases;

КОЕЗДВССОП_1.2.2 : Targets the security measures in the specific case where the instance of the software under investigation continues to run after a procedure of standard user-requested logout has been completed;

КОЕЗДВССОП_1.2.3 : Targets the security measures in the specific case where the user performs a FORCE STOP;

КОЕЗДВССОП_1.2.4 : Targets the security measures in the specific case where the instance of the software under investigation continues to run after a procedure of implicit logout / implicit vault lockout, triggered by the built-in auto-lock protection due to inactivity, has been completed;

КОЕЗДВССОП_1.2.5 : Targets the security measures in the specific case where the user removes the instance of the software under investigation from the system's screen of recently run applications immediately after a procedure of implicit logout / vault lockout, activated by the built-in auto-lock protection due to inactivity, has been completed;

МКОЕЗДВССОП_2 : Security measures for limiting exposure of software's own login data in main memory – Checks whether the software under investigation implements effective security measures to minimize the amount of its login data[1] exposed in cleartext in its owned areas in main memory. It is made of 2 sets, described in brief bellow:

МКОЕЗДВССОП_2.1 : Security measures for protecting the confidentiality of the software's user identity from the previous session – Checks whether the software under investigation takes adequate actions to protect the confidentiality of user identity, by implementing effective security measures to minimize the exposure of the login identifier for the user profile / vault in the main memory areas owned by the software. This set comprises 4 criteria, described in brief below:

КОЕЗДВССОП_2.1.1 : Targets the security measures in the specific case where the instance of the software under investigation continues to run after a procedure of standard user-requested logout has been completed;

КОЕЗДВССОП_2.1.2 : Targets the security measures in the specific case where the user performs a FORCE STOP;

КОЕЗДВССОП_2.1.3 : Targets the security measures in the specific case where the instance of the software under investigation continues to run after a procedure of implicit logout / implicit vault lockout, triggered by the built-in auto-lock protection due to inactivity, has been completed;

1 The login data for the software under investigation itself, the confidentiality breach of which can be exploited by a potential attacker to compromise the security of "ДД", which are subject to long-term storage in the vault.

КОЕЗДВССОП_2.1.4 : Targets the security measures in the specific case where the user removes the instance of the software under investigation from the system's screen of recently run applications immediately after a procedure of implicit logout / vault lockout, activated by the built-in auto-lock protection due to inactivity, has been completed;

МКОЕЗДВССОП_2.2 : **Security measures for protecting the confidentiality of ТУУ** – this set requires the software under investigation to take adequate actions to protect the confidentiality of **ТУУ**, by taking immediate actions to minimize their exposure in cleartext in main memory. This set comprises one criterion, described in brief below:

КОЕЗДВССОП_2.2.1 : HSS is allowed to load **ТУУ** in cleartext in main memory only if strictly necessary, with immediate action taken for secure deletion of the **ТУУ** as soon as the strict necessity ceases;

Identifier of:		Maximum Scores	Minimum Scores Required to Consider the Set / Criterion As		
Set	Criterion		Satisfied	Partially satisfied	Not Satisfied
МКОЕЗДВССОП_1		50	50	10	0
МКОЕЗДВССОП_1.1		32,5	32,5	6,5	0
	КОЕЗДВССОП_1.1.1	21,125	21,125	4,225	0
	КОЕЗДВССОП_1.1.2	3,4125	3,4125	0,6825	0
	КОЕЗДВССОП_1.1.3	1,1375	1,1375	0,2275	0
	КОЕЗДВССОП_1.1.4	3,4125	3,4125	0,6825	0
	КОЕЗДВССОП_1.1.5	3,4125	3,4125	0,6825	0
МКОЕЗДВССОП_1.2		17,5	17,5	3,5	0
	КОЕЗДВССОП_1.2.1	11,375	11,375	2,275	0
	КОЕЗДВССОП_1.2.2	1,8375	1,8375	0,3675	0
	КОЕЗДВССОП_1.2.3	0,6125	0,6125	0,1225	0
	КОЕЗДВССОП_1.2.4	1,8375	1,8375	0,3675	0
	КОЕЗДВССОП_1.2.5	1,8375	1,8375	0,3675	0
МКОЕЗДВССОП_2		50	50	10	0
МКОЕЗДВССОП_2.1		2,5	2,5	0,5	0
	КОЕЗДВССОП_2.1.1	0,75	0,75	0,15	0
	КОЕЗДВССОП_2.1.2	0,25	0,25	0,05	0
	КОЕЗДВССОП_2.1.3	0,75	0,75	0,15	0

	КОЕЗДВССОП_2.1.4	0,75	0,75	0,15	**0**
МКОЕЗДВССОП_2.2		47,5	47,5	9,5	**0**
	КОЕЗДВССОП_2.2.1	47,5	47,5	9,5	**0**
All together	**All together**	100	100	20	**0**

Table 2: Security Assessment Scale

As part of the dissertation work, a security assessment scale was proposed. The scale itself is presented in Table 2.

4.3.4. Usage Scenarios Definition

As part of the preparation for conducting the digital investigation, multiple specialized usage scenarios (ССУ) have been defined, based on the МКОЕЗДВССОП (from Section 4.3.3) and with the assistance of the template presented in Figure 7. In the dissertation, the definitions of the following scenarios, deemed most significant for preparing an assessment of the effectiveness of in-memory data protection in highly secure software, are provided:

ССУ_1 : Re-launching the software under investigation, immediately after FORCE STOP procedure has been completed

ССУ_2 : Completing the user identity verification process to have the vault’s home screen opened

ССУ_4 : Opening an existing record;

ССУ_5 : Opening an existing record having a file attachment;

ССУ_6 : Opening the text file attached to an existing record;

ССУ_7 : Triggering the vault’s auto-lock security feature;

ССУ_8 : Triggering the vault’s auto-lock security feature, followed by an explicit removal of software under investigation from the system's screen of recently run applications;

ССУ_9 : Logging out;

4.3.5. Digital Investigation Results and Findings

This section briefly describes the concrete digital investigation conducted, where the efficiency of in-memory data protection in software under investigation (selected in Section 4.3.1) is evaluated. It also analyzes the results from security standpoint.

${идентификатор_на_група_от_моментни_снимки} ⇠ ${идентификатор_на_сценарий}

Figure 10: Notation Used to Denote the Result of Inspection Results Report for Software Under Investigation

The notation used to support the process of describing the results is shown in Figure 10. It is used to denote the findings of inspection report for the software under investigation.

4.3.5.1. *Keeper Results Analysis*

In Figure 11 and Figure 12, the results are presented in graphical form based on the inspection of the designated groups of memory dumps for Keeper's private areas in main memory. The complete analysis of the results, based on the inspection report for Keeper, is described in the dissertation.

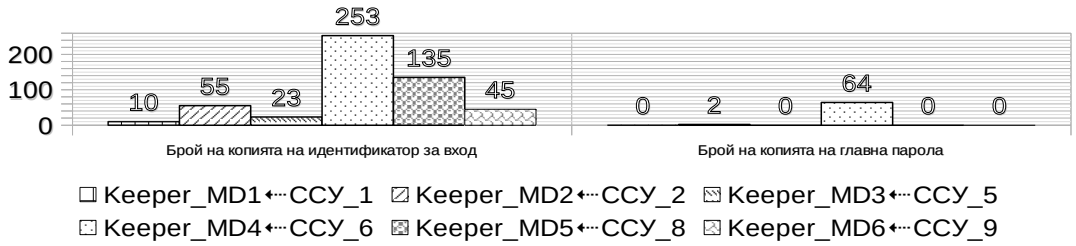

Figure 11: Number of copies of the values exposed in cleartext in the main memory from the group "Data for unlocking the vault of the investigated software", based on the inspection results report for Keeper.

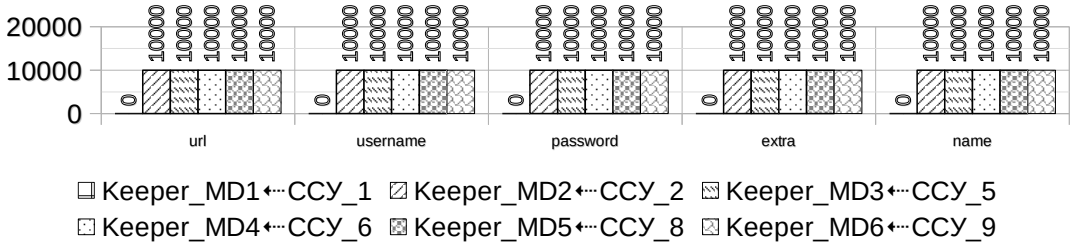

Figure 12: Number of records for which at least one copy of a non-empty and non-zero field value from the group „ДД representing login data for external sites, stored in the vault of the investigated software" is exposed in cleartext in the main memory, based on the inspcetion results report for Keeper.

4.3.5.2. *Bitwarden* Results Analysis

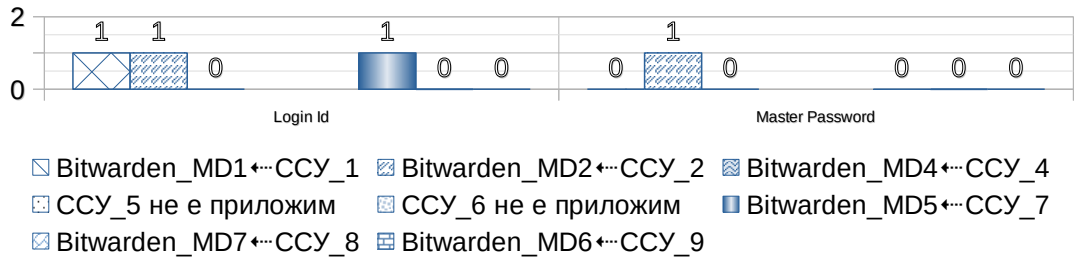

Figure 13: Number of copies of the values exposed in cleartext in the main memory from the group "Data for unlocking the vault of the investigated software", based on the inspection results report for Bitwarden

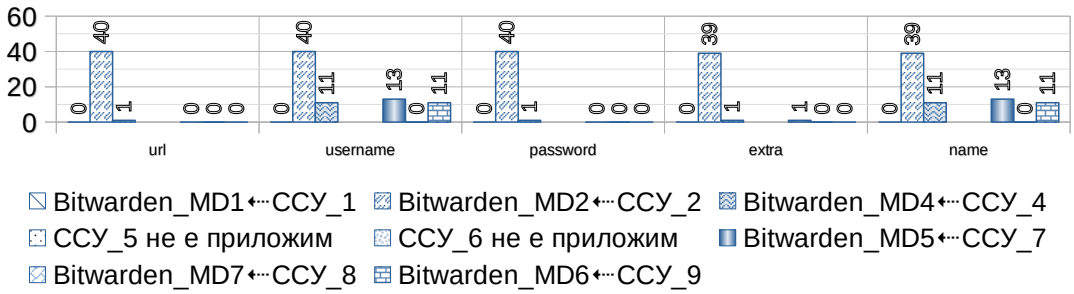

Figure 14: Number of records for which at least one copy of a non-empty and non-zero field value from the group „ДД representing login data for external sites, stored in the vault of the investigated software" is exposed in cleartext in the main memory, based on the inspcetion results report for Bitwarden

In Figure 13 and Figure 14, the results are presented in graphical form based on the inspection of the designated groups of memory dumps for Bitwarden's private areas in main memory. The complete analysis of the results, based on the inspection report for *Bitwarden*, is described in the dissertation.

4.3.6. Security Evaluation With Regard to the Efficiency of Data Protection in Main Memory Areas Belonging to the Selected Representatives of Existing Password Management Software / Software Digital Vault.

Identifier of		Keeper		Bitwarden	
Set	**Criterion**	**Security Rating**	**Security Score**	**Security Rating**	**Security Score**
МКОЕЗДВССОП_1		Not Satisfied	1,75	Partially satisfied	49,871346898673
МКОЕЗДВССОП_1.1		Not Satisfied	1,1375	Partially satisfied	32,416385625
	КОЕЗДВССОП_1.1.1	Not Satisfied	0	Partially satisfied	21,04155625
	КОЕЗДВССОП_1.1.2	Not Satisfied	0	Satisfied	3,4125
	КОЕЗДВССОП_1.1.3	Satisfied	1,1375	Satisfied	1,1375
	КОЕЗДВССОП_1.1.4	Not Satisfied	0	Partially satisfied	3,412329375
	КОЕЗДВССОП_1.1.5	Not Satisfied	0	Satisfied	3,4125
МКОЕЗДВССОП_1.2		Not Satisfied	0,6125	Partially satisfied	17,454961273673
	КОЕЗДВССОП_1.2.1	Not Satisfied	0	Partially satisfied	11,332901273673
	КОЕЗДВССОП_1.2.2	Not Satisfied	0	Partially satisfied	1,8361525
	КОЕЗДВССОП_1.2.3	Satisfied	0,6125	Satisfied	0,6125
	КОЕЗДВССОП_1.2.4	Not Satisfied	0	Partially satisfied	1,8359075
	КОЕЗДВССОП_1.2.5	Not Satisfied	0	Satisfied	1,8375
МКОЕЗДВССОП_2		Not Satisfied	0	Not Satisfied	1,5
МКОЕЗДВССОП_2.1		Not Satisfied	0	Partially satisfied	1,5
	КОЕЗДВССОП_2.1.1	Not Satisfied	0	Satisfied	0,75
	КОЕЗДВССОП_2.1.2	Not Satisfied	0	Not Satisfied	0
	КОЕЗДВССОП_2.1.3	Not Satisfied	0	Not Satisfied	0
	КОЕЗДВССОП_2.1.4	Not Satisfied	0	Satisfied	0,75
МКОЕЗДВССОП_2.2		Not Satisfied	0	Not Satisfied	0
	КОЕЗДВССОП_2.2.1	Not Satisfied	0	Not Satisfied	0
All together	**All together**	Not Satisfied	1,75	Partially satisfied	51,371346898673

Table 3: Security Evaluation of the Effectiveness of Data Protection in the Software Under Investigation in Its Areas in Main Memory, Based on the Results of the Digital Investigation and the МКОЕЗДВССОП (Defined in Section 4.3.3)

In Table 3, an evaluation of the effectiveness of data protection in the software under investigation in its areas in main memory is presented. This evaluation is made according to the criteria specifically defined in Section 4.3.3, based on the results of the conducted digital investigation. It serves as a security assessment report for the effectiveness of in-memory data protection in the context of software under investigation.

As seen in Figure 15, the *Keeper* software failed to meet the lower bound for partial satisfaction of any of the sets defined in Section 4.3.3 (МКОЕЗДВССОП), which in turn means that the in-memory data protection[2] provided by *Keeper* is extremely ineffective. Unlike *Keeper*, however, the *Bitwarden* software performed significantly better, achieving a total score for all sets:

2 In terms of data protection in its owned areas in main memory.

51.371346898673, which is significantly higher compared to *Keeper*'s total score for all sets: 1.75 and completely refutes the officially made claims that "*Keeper* is the most secure password manager app for *Android*!" [32] and "Most Secure Password Manager in the Industry" [32]. Also, despite the data protection provided by *Bitwarden* in its areas in main memory being assessed as significantly more effective compared to that provided by *Keeper*, *Bitwarden* fails to meet the lower bound for partial satisfaction of МКОЕЗДВССОП_2 and МКОЕЗДВССОП_2.2 .

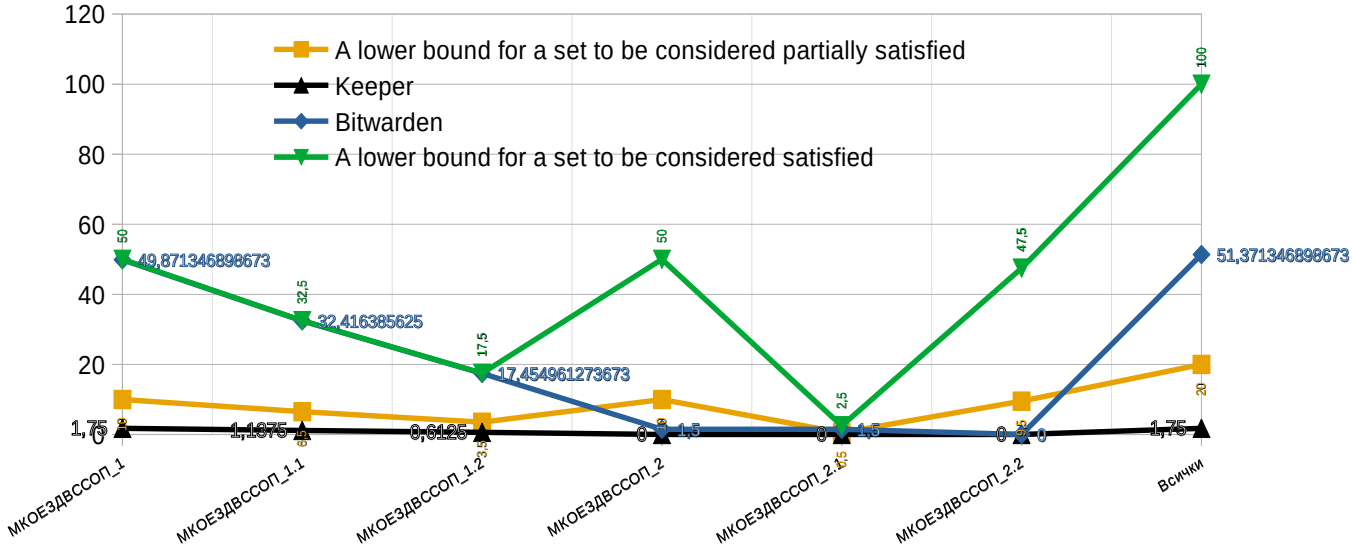

Figure 15:Security Comparison in Terms of Effectiveness of Data Protection in the Software Under Investigation in Its Areas in Main Memory, Based on the Results of the Digital Investigation and the МКОЕЗДВССОП (Defined in Section 4.3.3)

4.4. Conclusion and Key Findings

In this chapter, a main memory inspection methodology with regard to the efficiency of in-memory data protection in the context of HSS execution is defined. A specialized software tool for detecting and extracting cleartext data exposed in main memory has been designed and developed. Subsequently, the applicability of the defined methodology and the developed specialized software tool was validated by using them to conduct a digital investigation to security evaluate the effectiveness of in-memory data protection in the context of representative sample HSS execution. Specifically, in accordance with the methodology's prescriptions, two representative applications were selected, and immediately after their selection, an experimental device and a sufficiently large set of uniquely identifiable data for the purposes of the experiments were prepared. Specific security evaluation criteria with regard to the efficiency of in-memory data protection in HSS, as well as specialized usage scenarios to support the conduct of the experiments, were defined. A digital investigation was then conducted to examine the effectiveness of in-memory data protection in the context of executing the specific HSS from the representative sample. An analysis of the results obtained from the digital investigation was performed, and an assessment of the effectiveness of the protection of the trusted data of the software from the representative sample in its areas in main memory was prepared. It has been found that there are several gaps concerning in-memory data protection in the context of the software under investigation. Specifically, for Keeper, it was found out that the data protection provided in Keeper's areas in main memory tends to be extremely ineffective, as there is a sufficiently large time window to allow for the relatively easy theft of the master password and the trusted data meant to be securely stored. This is because Keeper keeps all trusted data loaded in main memory in cleartext for the entire duration of the Keeper process and its dependent services, even when the user has explicitly logged out. In contrast, the Bitwarden software was found to provide significantly more effective data protection in its main memory areas than Keeper. However, significant gaps were also identified for Bitwarden regarding the effectiveness of in-memory data protection. Like Keeper, Bitwarden has failed to meet the lower bound for partial satisfaction of МКОЕЗДВССОП_2 and МКОЕЗДВССОП_2.2. It allowed for some of the trusted data to remain in main memory in cleartext even when the user has explicitly logged out.

The following conclusions can be drawn: 1) every HSS should consider the specifics of the execution environment and potential side effects related to securely data deletion in the context of

automatic memory management; 2) every HSS should take necessary measures to ensure the confidentiality, authenticity, and integrity of data in main memory, including applying strategies for immediate encryption of sensitive data in main memory, as well as minimizing the dwell time of such data in main memory, and in addition, undertake specific actions (secure erasure of sensitive data immediately after use, applying page-based memory locking techniques, disabling memory dumps, ...) to ensure that data in the address space of the executing software instance in main memory will not leak from there or be stolen.

Chapter 5: Conclusion, Contributions, and Plans for Future Development

5.1. Summary of the Achievement of the Dissertation Aim and Objectives

In the introductory part of the dissertation, the necessity of data protection is motivated. The concept of highly secure software and related concepts, definitions, and terms are introduced, with special attention given to the specific type of specialized HSS: PMS / DVS. The work on the dissertation continues with a review based on which an analysis of the current state in the researched field is performed. As part of this review, problems related to data security, security requirements, and data security threats are examined and analyzed, while numerous scientific papers and books in the field are reviewed and analyzed. The main threats to data security for HSS of any type are identified and subsequently defined, and basic terms and definitions supporting the planned further research and experiments in the dissertation are introduced. One fundamental requirement for data security and protection (ОИСЗД) is elicited and specified. A study of various scientific papers, recommendations, guidelines, approaches, principles, standards, and data protection measures applicable for supporting the design and implementation of HSS is conducted. Based on this study, an analysis is performed, and cryptanalysis and databases with known vulnerabilities are considered, filtering out secure from insecure approaches, principles, techniques, standards, and protection measures. In addition, cryptographic protection measures are classified into two main groups. Attention is also given to tools supporting the conduct of digital investigations, security analysis, detection of potential vulnerabilities / breaches in security, and reverse engineering activities. A methodology for conducting an investigation, comparative security analysis and in breadth security evaluation of HSS at a high-level is defined, and subsequently, the applicability of the defined methodology is validated by applying it to study, security compare and in breadth security evaluate a representative sample of PMS / DVS according to specific security evaluation criteria. Specifically, based on the results from Chapter 3, a representative sample of two software applications is selected, and immediately after the selection, necessary actions are taken to prepare an experimental device and a sufficiently large set of uniquely identifiable data. Specific security evaluation criteria with regard to the efficiency of in-memory data protection in HSS, as well as specialized usage scenarios to support the conduct of the experiments, are defined. A digital investigation is then conducted to examine the effectiveness of in-memory data protection in the context of executing the specific HSS from the representative sample. An analysis of the results obtained from the digital investigation is performed, and an assessment of the effectiveness of the protection of the trusted data of the software from the representative sample in its areas in main memory is prepared.

In conclusion, the main goal of the dissertation, namely to support the security research of PMS / DVS, including to security analyze and security evaluate this type of highly secure software both generalized (in breadth, at high level) and with respect to the data security and protection effectiveness while PMS / DVS is running into the main memory, including the security measures the PMS / DVS is taking in order to limit sensitive data exposure in its own main memory areas, has been successfully achieved. All initially set research objectives and tasks have also been successfully completed.

5.2. Contributions

The key contributions of this thesis are as follows:

Scientific Contributions:

- A methodology for conducting an investigation, comparative security analysis and in breadth security evaluation of HSS at a high-level is defined; (Chapter 3)
- A main memory inspection methodology with regard to the efficiency of in-memory data protection in the context of HSS execution is defined; (Chapter 4)

Scientific and Applied Contributions:

- A process has been proposed for applying sets of security evaluation criteria, with the aim of using it for comparative analysis and evaluation of the security of existing HSS of the PMS/DVS type; (Chapter 3)
- The applicability of the methodology for conducting an investigation, comparative security analysis and in breadth security evaluation of HSS at a high-level has been validated by applying it to study, security compare and in breadth security evaluate a representative sample of PMS / DVS; (Chapter 3)
- Specific security evaluation criteria with regard to the efficiency of in-memory data protection in HSS, as well as specialized usage scenarios to support the conduct of the experiments, are defined; (Chapter 4)
- A digital investigation is conducted to examine the effectiveness of in-memory data protection in the context of HSS execution, then an analysis of the results obtained is performed, and a security evaluation with regard to the effectiveness of data protection in main memory is prepared; (Chapter 4)
- The applicability of the main memory inspection methodology with regard to the efficiency of in-memory data protection in the context of HSS execution is validated by applying it to study, security compare and in breadth security evaluate a representative sample of PMS / DVS according to specific security evaluation criteria; (Chapter 4)

Applied Contributions:

- A specialized software tool for supporting the conduct of digital investigation, analysis and security evaluation regarding the effectiveness of data protection in main memory in the context of HSS execution has been designed and developed; (Chapter 4)

5.3. Publications

List of publications on the topic of the dissertation:

- *Peter Sabev, Milen Petrov*, **Securing User Data in Android Applications at Software Level**, Computer & Communications Engineering, vol:13, issue:2, 2019, pages:44-47, ISSN (print):1314-2291, PhD
- *Peter Sabev, Milen Petrov*, **Android Password Managers and Vault Applications: Comparative Security Analysis**, 2021 International Conference Automatics and Informatics (ICAI), 2021, pages:198-205, ISSN (print): 978-1-6654-2662-6, ISBN: 978-1-6654-2661-9, doi:10.1109/ICAI52893.2021.9639693, Ref, IEEE Xplore (https://ieeexplore.ieee.org/document/9639693), PhD
- *Peter Sabev*, *Milen Petrov*, **Android Password Managers and Vault Applications: An Investigation on Data Remanence in Main Memory**, ISGT 2021 Information Systems and Grid Technologies, 2021, pages:314-328, ISSN (online):1613-0073, Ref, SCOPUS, SJR (0.18 -)
- *Peter Sabev, Milen Petrov*, **Chapter Eight: Requirements for Securing User Data in Android Applications at Software Level**, 2021, ISBN:1-5275-6066-X, Cambridge Scholars Publishing, Cambridge Scholars Publishing,

Conference reports:

- Sectional report, *Peter Sabev*, **Securing User Data in Android Applications at Software Level**, 2019, Workshop on Information Security (BCI 2019)
- Sectional report, *Peter Sabev*, **Android Password Managers and Vault Applications: Comparative Security Analysis**, 2021, ICAI2021 (International Conference Automatics and Informatics)
- Sectional report, *Peter Sabev*, **Android Password Managers and Vault Applications: An Investigation on Data Remanence in Main Memory**, 2021, ISGT'2021 (14th Information Systems and Grid Technologies Conference)

5.4. Plans for Future Development

After conducting a study, comparative analysis, and broad security evaluation of existing PMS/DVS, further research in the dissertation focused primarily on one of the most critical aspects of security for PMS/DVS, namely the security aspect about securing the software's data in its own main memory areas and limiting the exposure of sensitive data in cleartext there. However, besides this aspect, there are other key security aspects, such as the aspect related to the protection of trusted data for their secure long-term storage, which was not deeply explored in the dissertation, mainly due to considerations aiming to avoid over-expanding the scope of the dissertation. Given this, a future in-depth research regarding the aspect of protecting trusted data for their secure long-term storage is planned. In addition, in the digital investigation conducted as part of the dissertation regarding the security aspect about securing the software's data in main memory, to avoid over-expanding the scope of the dissertation, the investigation of areas in main memory belonging to third-party software processes, the operating system, shared buffer memory, and other temporary memories like CPU/GPU caches and their technical specifics, including other areas in operational memory/other types of temporary memory designated for access by low-level software, was entirely skipped. However, such a type of investigation is also particularly significant from a security standpoint, for which it would be beneficial to be planned for future conduct.

Regarding the fact that none of the investigated software applications (from the representative sample) managed to fully satisfy all sets of security assessment criteria, including the Bitwarden software, which was rated as the most secure and with the highest level of transparency: 1) It is advisable for all identified security related architectural errors to be systematically described as anti-patterns with accompanying explanations regarding the nature of the errors and analysis of the potential risks/consequences of their use in terms of data security; 2) A conclusion can be made that it is necessary to plan actions for developing new architectural solutions, based on which it would be possible to build a new HSS of the PMS/DVS type, capable of achieving satisfaction (to a full extent) of all sets of security evaluation criteria.

Bibliography

[1] C. Herley and P. Van Oorschot, “A Research Agenda Acknowledging the Persistence of Passwords,” *IEEE Secur. Priv. Mag.*, vol. 10, no. 1, pp. 28–36, Jan. 2012, doi: 10.1109/MSP.2011.150.

[2] S. Ruoti, J. Andersen, and K. Seamons, “Strengthening Password-based Authentication,” in *SOUPS 2016: Twelfth Symposium on Usable Privacy and Security*, 2016. Accessed: Jan. 15, 2023. [Online]. Available: https://www.usenix.org/conference/soups2016/workshop-program/way2016/presentation/ruoti_password

[3] P. A. Grassi *et al.*, “Digital identity guidelines: authentication and lifecycle management,” Gaithersburg, MD, Jun. 2017. doi: 10.6028/NIST.SP.800-63b.

[4] M. Papathanasaki, L. Maglaras, and N. Ayres, “Modern Authentication Methods: A Comprehensive Survey,” *AI, Comput. Sci. Robot. Technol.*, vol. 2022, pp. 1–24, Jun. 2022, doi: 10.5772/acrt.08.

[5] J. Lam, “Security Intentions and the Persistence of Passwords,” 2022. Accessed: Jan. 15, 2023. [Online]. Available: https://bitwarden.com/images/security-intentions-and-the-persistence-of-passwords.pdf

[6] G. A. Miller, “The magical number seven, plus or minus two: some limits on our capacity for processing information,” *Psychol. Rev.*, 1956, doi: 10.1037/h0043158.

[7] G. J. Johnson, “A distinctiveness model of serial learning.,” *Psychol. Rev.*, 1991, doi: 10.1037//0033-295x.98.2.204.

[8] Verizon, “2018 Data Breach Digest,” 2019. Accessed: Feb. 01, 2023. [Online]. Available: https://www.verizon.com/business/resources/reports/2018-data-breach-digest.pdf

[9] I. Ion, R. Reeder, and S. Consolvo, “‘...No one can hack my mind’: Comparing expert and non-expert security practices,” in *SOUPS 2015 - Proceedings of the 11th Symposium on Usable Privacy and Security*, 2015.

[10] “Operating System Market Share Worldwide | Statcounter Global Stats.” https://gs.statcounter.com/os-market-share#yearly-2019-2022 (accessed Jan. 20, 2023).

[11] “Scopus - Analyze search results (TITLE-ABS-KEY (data AND security) AND ...” https://www.scopus.com/term/analyzer.uri?sid=664dd2ef3b759abdeb9a3f226d5bb6aa&origin=resultslist&src=s&s=TITLE-ABS-KEY%2528data+security%2529&sort=r-f&sdt=cl&sot=b&sl=28&count=224172&analyzeResults=Analyze+results&cluster=scopubyr%252C%25222022%2522%252Ct%252C%25222021%252 (accessed Jan. 28, 2023).

[12] R. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems, 3rd Edition*, Third. Wiley, 2021.

[13] S. Jacobs, *Engineering Information Security: The Application of Systems Engineering Concepts to Achieve Information Assurance*. Wiley-IEEE Press, 2015.

[14] I. Sommerville, *Software Engineering, 10th Edition*. Pearson, 2016. Accessed: Feb. 02, 2023. [Online]. Available: www.pearsonglobaleditions.com

[15] B. Schneier, *Applied Cryptography: Protocols, Algorithms and Source Code in C, 2nd Edition*. Wiley, 2015.

[16] A. Canteaut, “Stream Cipher,” in *Encyclopedia of Cryptography and Security*, Boston, MA: Springer US, 2011, pp. 1263–1265. doi: 10.1007/978-1-4419-5906-5_374.

[17] E. Barker, “SP 800-57 Part 1 Rev. 5, Recommendation for key management: Part 1 – General,” *NIST Spec. Publ. 800-57*, May 2020, doi: 10.6028/NIST.SP.800-57pt1r5.

[18] ENISA, “Algorithms, key size and parameters report – 2014,” 2014. https://www.enisa.europa.eu/publications/algorithms-key-size-and-parameters-report-2014 (accessed Oct. 05, 2020).

[19] “EPC342-08 / Version 11.0: Guidelines on cryptographic algorithms usage and key management.” 2022. Accessed: Feb. 12, 2023. [Online]. Available: https://www.europeanpaymentscouncil.eu/document-library/guidance-documents/guidelines-cryptographic-algorithms-usage-and-key-management-0

[20] J. C. S. Santos, K. Tarrit, and M. Mirakhorli, “A Catalog of Security Architecture Weaknesses,” in *2017 IEEE International Conference on Software Architecture Workshops (ICSAW)*, Apr. 2017, pp. 220–223. doi: 10.1109/ICSAW.2017.25.

[21] D. A. Wheeler, “What is Open Security?” 2013. Accessed: Jul. 16, 2023. [Online]. Available: https://apps.dtic.mil/sti/pdfs/ADA607073.pdf

[22] J. H. Saltzer and M. D. Schroeder, “The protection of information in computer systems,” *Proc. IEEE*, vol. 63, no. 9, pp. 1278–1308, 1975, doi: 10.1109/PROC.1975.9939.

[23] “Android keystore system.” https://developer.android.com/training/articles/keystore.html (accessed Mar. 06, 2023).

[24] “Keeper Encryption Model.” https://docs.keeper.io/enterprise-guide/keeper-encryption-model (accessed Oct. 05, 2020).

[25] “Keeper End-User Guid - Android - Settings.” https://docs.keeper.io/user-guides/android#settings (accessed Oct. 05, 2020).

[26] “LastPass Technical Whitepaper.” https://assets.cdngetgo.com/aa/97/5ec2619b46349dc3eb3212a37b11/lastpass-technical-whitepaper.pdf (accessed Oct. 05, 2020).

[27] “Dashlane Security White Paper,” 2019. https://www.dashlane.com/download/Dashlane_SecurityWhitePaper_October2018.pdf (accessed Oct. 05, 2020).

[28] “Help Center | Bitwarden Help & Support,” 2020. https://bitwarden.com/help/ (accessed Oct. 05, 2020).

[29] “Bitwarden Crypto.” https://bitwarden.com/help/crypto.html (accessed Oct. 05, 2020).

[30] “Bitwarden Security Assessment Report,” 2018. https://cdn.bitwarden.net/misc/Bitwarden Security Assessment Report.pdf (accessed Oct. 05, 2020).

[31] “Bitwarden - Repositories.” https://github.com/bitwarden (accessed Oct. 05, 2020).

[32] “Password Manager - Keeper - Google Play.” https://play.google.com/store/apps/details?id=com.callpod.android_apps.keeper&hl=en_US (accessed Oct. 05, 2020)

Declaration of originality of results

I hereby declare that this dissertation contains original results obtained by me with the support and assistance of my supervisors. The results obtained by other scientists are described in detail and cited in the bibliography.

This dissertation has not been applied for the acquisition of an educational and scientific PhD in another school, university, or scientific institute.

Signature:……...