**OFFLINE MODE FOR MOBILE CLIENT SECURITY ARCHITECTURE: A PRACTICAL APPROACH**

(suggest your titles) **MOBILE CLIENT SECURITY ARCHITECTURE: A PRACTICAL APPROACH**

T. GALIBUS1, V. KRASNOPROSHIN1

1*Belarusian State University, Minsk, 4, Nezavisimosti, 220030, Belarus*

THIAGO P.B. VIEIRA2, ROBSON DE OLIVEIRA ALBUQUERQUE2, RAFAEL TIMÓTEO DE SOUSA JÚNIOR2, JOÃO PAULO C. L. COSTA2

2*University of Brasilia, UnB - FT – ENE – CP: 4386, 70910-900, Brasília - DF – Brazil*

EDISON PIGNATON DE FREITAS3

3*Federal University of Rio Grande do Sul, UFRGS – INF – CP: 15064, 91501-970 Porto Alegre - RS - Brazil*

ANTON ZALESKI4, H.E.R.M. VISSIA4

*4Byelex Multimedia Products BV Argon 1, 4751 XC Oud Gastel, The Netherlands*

**Acknowledgement – I should add the CAPES project here, can anyone provide the text?**

**Abstract:**

This paper presents a novel approach to the security of the corporate mobile devices, in particular the offline mode. The important part of the presented approach is the conceptualization and the definition of the core methodology to solve the problem of offline mobile security, i.e. the protection of the confidential data in use when the mobile client is not connected to the corporate cloud. The protection of the sensitive data is provided by the combination of the cryptographic methods, such as AES file encryption, ABE authorization based both on user and share attributes, shared key SSS-based protection between the device and the user as well as MOS-based analytics methods to prevent the malicious user behavior. The proposed security architecture supports the basic mobile device protection principles: minimized traffic load and reduced communication with the cloud; use of the light-weighted operations and an optimized combination of the security methods.

Keywords: mobile security, offline mode, ABE, MOS, user behavior analysis, secret sharing, protected cloud

1. Introduction

Cloud computing is a new rapidly evolving paradigm in the world of distributed networking and computation. The basic features of the cloud environment is providing the elastic, on-demand and secure services for the end-users. While the first two requirements are rather well conceptualized and supported by the majority of the cloud platforms in use, security is a serious concern of the cloud providers and governmental organizations as well as academia and research community [1], [2], [3]. For the small and medium-sized enterprises (SME) the cloud environment is often the most cost-effective and easily scalable solution. However, the security and privacy of the sensitive data in the cloud is not so obvious. It is also not fully conceptualized, thus it leads to obscure and incomplete security paradigms and solutions.

A common practice to provide a stable security cloud solution is to use a specific type of cloud service: CASB (Cloud Access Security Broker) or CAC (Cloud Access Control). These services are specifically designed to bring security at a single access point and provide the coordination of the most important security measures. It is estimated by Gartner [4] that such systems will be used by 85% of companies by 2020. Obviously, the reason for this is that the organization of security measures at a single control point allows to control and to monitor the level of cloud protection much more effectively. The basic features of the CASB are: discovery of cloud services, encryption (along with tokenization for better search properties), access control, DLP (Data Loss Prevention) services, authentication and auditing/alerting services [5]. The protection of the confidential data, according to the standards of CASB deployment should be provided elsewhere, i.e. in transit and at the end-user (client) side [6].

Additional security issues and requirements have to be considered when mobile devices are actively used in corporate cloud environment [7]. Today more and more organizations and enterprises are functioning in the BYOD (Bring-Your-Own-Device) paradigm. The uncontrolled usage of the mobile devices represents a serious risk to the development of secure SME cloud platforms being the bottleneck of the corporate information security system (ISS). While the enterprise cloud infrastructure based on the web interface can be protected by powerful third-party services, such as CASB and CAC, the corporate mobile app is usually light-weighted and generally less protected. The protection scheme used on a mobile device should be both computationally secure as well as resource-constrained due to battery power limitations [8]. Therefore, encrypting files and generating keys on a mobile device is not considered a good solution. On the other hands, the protection schemes with good computational qualities lack the security analysis in many cases [9]. The common practice is the shadow user activities monitoring [7]. However, the mobile device usage stays unprotected in all the proposed scenarios while in offline mode.

Suppose, a SME uses CASB in order to protect data at rest (i.e. while stored on the server-side), in transit (communication with server), in use (while the client is connected to the network). But when the mobile client goes offline with the sensitive corporate data on board all powerful cloud-based tools cannot help and the mobile app has to secure itself with its own limited resources. Moreover, due to the resources constraint there is a crucial difference in strategy of online and offline mode protection. For example, offline mode does not allow performing the extensive computation and encryption on the mobile device.

Observing the above described landscape, this paper outlines the concept of the offline mobile client security. The proposal contains a novel approach based on powerful cryptographic preventive methods, such as secret sharing [10] and ABE encryption [11]. Also, the proposal includes the usage of the user behavior analysis based on Model Order Selection (MOS) [12], in order to highlight possible threats, reduce the risks and the harm of the most common threats: the expired user misusing password and the intruder attack. The key expiration period is safely incorporated into the proposed system solution in order to enhance security. Additionally, the behavioral analysis can indicate well known malicious behaviors, their variations, as well as novel attacks, which present low or high variance in comparison to legitimate user behaviors. The main target of our proposed solution is to provide a maximum defense at the minimal resource cost.

The paper is structured as follows. Section 2, analyzes the most common security problems in the mobile cloud environment and the proposed solutions for them related to the offline protection in the BYOD world. Section 3 outlines the basic processes and the module security infrastructure of the proposed solution as well as the outline of online mode security. Section 4 presents the detailed scheme of the proposed solution to the problem of offline mobile client security. Section 5 provides the schemes of the main security blocks including encryption and key usage hierarchies as well as explanations of the core security methods: SSS, ABE and MOS. Section 6 discusses the security proofs including the outline of adversary model, common threat scenarios and the proposed analytics to discover abnormal usage. Also, it contains the complexity analysis and the practical implementation results. Section 7 concludes the paper.

**2. Related Work**

Obviously, the BYOD world demands more sophisticated data protection services compared to ordinary computing environment. A common practice is to provide additional contextual methods apart from authentication, DLP services and encryption (at rest, in transit, in use) [7, 8, 13, 14]. The contextual methods increase the security of the app at a maximum level with minimum resource requirements. The most commonly used are:

1. Using geolocation of the device to trace its usage;
2. Setting up the expiration period of an app ;
3. Secured transfer politics between apps;
4. Restricting access to the corporate app;
5. Setting up the expiration period of app pass/pin;
6. Setting up the counter of failed tries;
7. Restricted or prohibited offline access;
8. Logging and auditing.

Preventing data leakage on the mobile device is an important security problem. Therefore, administrators should take additional control and protection measures for the confidential data on the mobile device that leaves the boundaries of the organization. Generally, the most sensitive and confidential data should not be permitted to be transferred to the mobile device. However, what if the SME needs to allow the workers to work on such devices and even use them in the offline mode for the convenience and traffic reduce or even for a particular characteristic of the app or the business itself?

From the theoretical point of view of this problem, several surveys can be considered [7, 8, 13, 14]. The common point of all those surveys is the mobile cloud computing is a rapidly developing paradigm that poses many security and complexity problems. This type of systems requires new models of application and new way of using data storage services [8, 14]. An analysis of the existing models of mobile cloud computing is presented in [8, 14]. Commonly, all the models and protection schemes concentrate on the encryption properties and either perform the computations on their own [15, 16] or use the cloud provider to off-load the cryptographic operations [17, 18]. Obviously, it is natural the mobile device cannot handle all operations securely without assistance of a cloud provider, due to resources constraint and battery limitation.

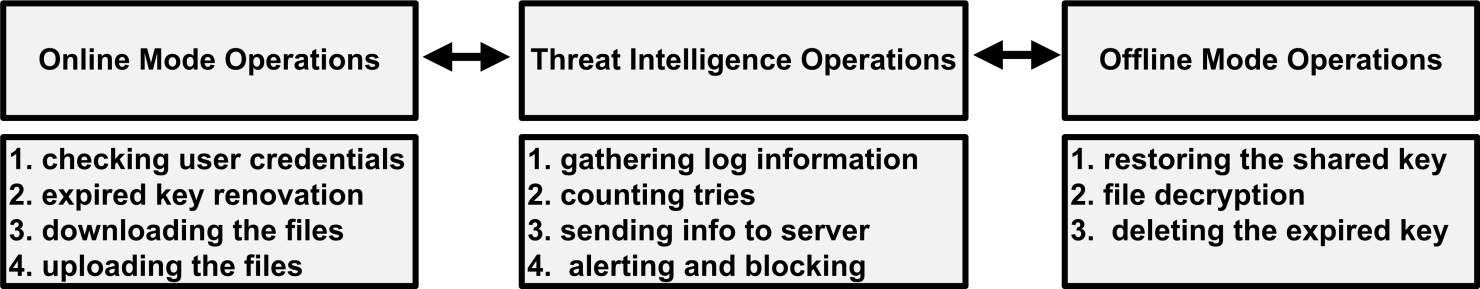
The necessity to use schemes that function without putting load on a provider arises when it is desired to make the device less dependent on the cloud provider i.e. corporate app continues to provide the secure access to the sensitive data without connection to the network. All the currently known schemes of encryption, performing the computation, as it is discussed in [8], either use a cloud provider [15], a third party trusted agent [16] or a combination of both [17]. In some cases, they concentrate on computational complexity without taking care of user privacy and security [18].

Therefore, from the overview of the surveys [8, 12, 14] it is concluded that the well-known mobile cloud security models proposed so far, do not consider the problem of the offline security mode. In many cases the industrial providers of the mobile security API avoid the problem by completely forbidding the offline access to the protected app, i.e. SAP Mocona, which operates as a secure app wrapping layer [19].

In some cases, the SME business procedure requires such access due to various constraints such as traffic load, travelling, ease of access and many more. So far and to the best of our knowledge the offline mode security problem has not yet been deployed, neither in academia nor in the industry [7, 8, 13, 14]. Therefore, the main concern of this proposal is the protection of the device/app in offline mode when the functions of data protection cannot be offloaded to a cloud or a trusted party.

**3. The client security architecture**

The approach proposed in this work describes and implements the complete lifecycle of the mobile app security infrastructure. The security processes depend on the key expiry period, which are used to access the protected storage. Once the user keys expire, the user is requested to enter his valid credentials, i.e. PIN and password. The client app then sends the credentials to the server for verification. Once the new set of access keys is received, the user can access the protected files in the offline mode, without the access to the server (this means no further communication with the server is needed until the key expires). The core set of functions and protocols can be divided into three processes as it is shown in the Figure 3.1:

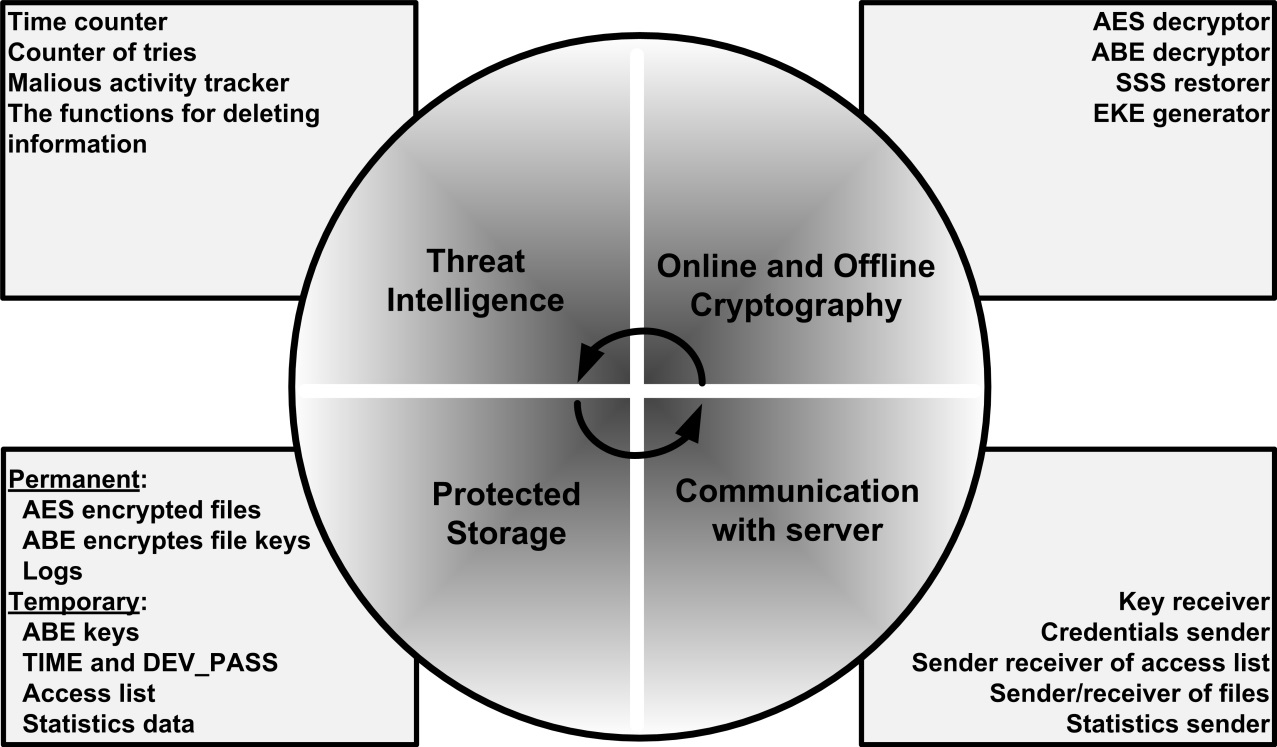


*Figure 3.1.*

The mobile client architecture consists of the following modules:

* *Cryptographic functions:* include the decryption (for AES and ABE) key restoration for SSS and EKE token generation procedure. While the AES-ABE hybrid encryption scheme functions both in the online and offline modes, the SSS is used only in the offline mode and the EKE generation and executions is used in the online mode. These functions will be described in the sections 5.1-5.4.
* *The threat intelligence infrastructure:* includes simple actors such as the time counter for the key expiry period, the counter of unsuccessful tries in order to protect from brute force attacks, and more elaborate MOS-inspired statistics analyzer. Functions such as alerting and deleting the expired key belong to this block as well. These functions will be described in the section 5.5
* *The communication with server*: includes the separate sender and receiver to check the user credentials and receiver new keys which acts based on the EKE protocol. The remaining data like access list and files can be sent via unprotected channels. Section 3.1 briefly discusses the communication with the server, but it is also an intention to make it a subject of our next work.
* *The storage:* it can be divided in two parts – temporary and permanent. While the files and file keys are stored permanently in order to reduce unnecessary traffic and the resource usage on client, the ABE keys and the key storage protection data are temporary. The unprotected files stored in the mobile client app memory are not the subject of the security architecture.

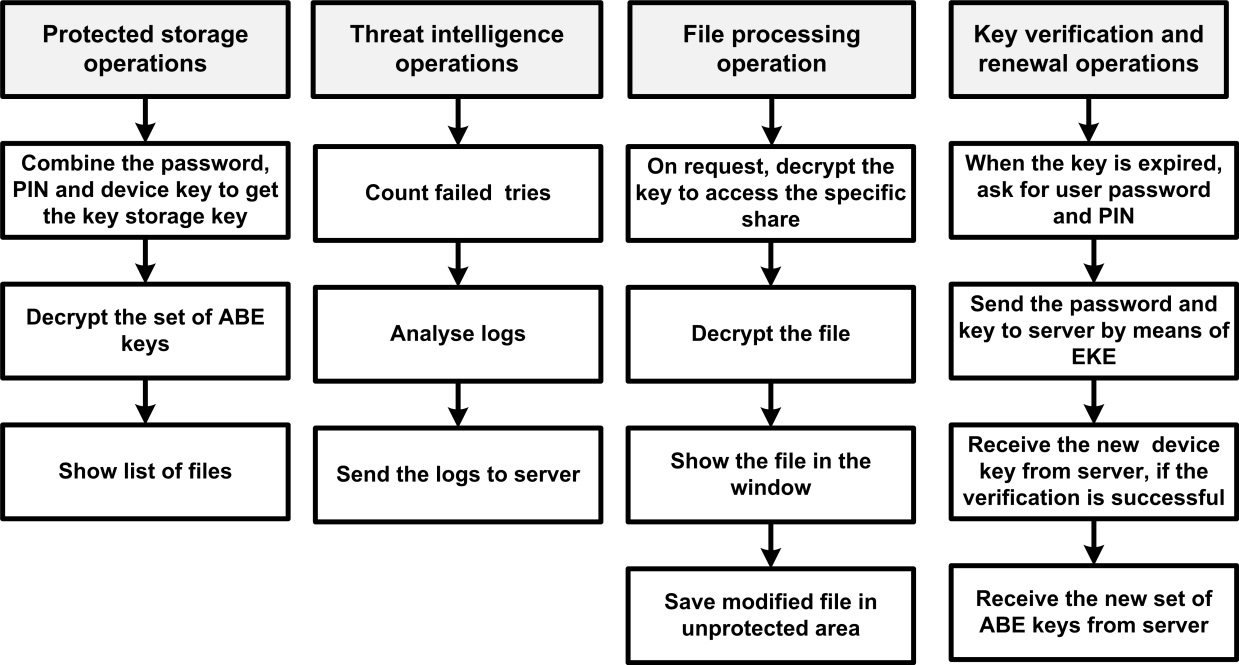
The figure 3.2 illustrates the client security architecture:



*Figure 3.2: Mobile security infrastructure*

**3.1. Online mode**

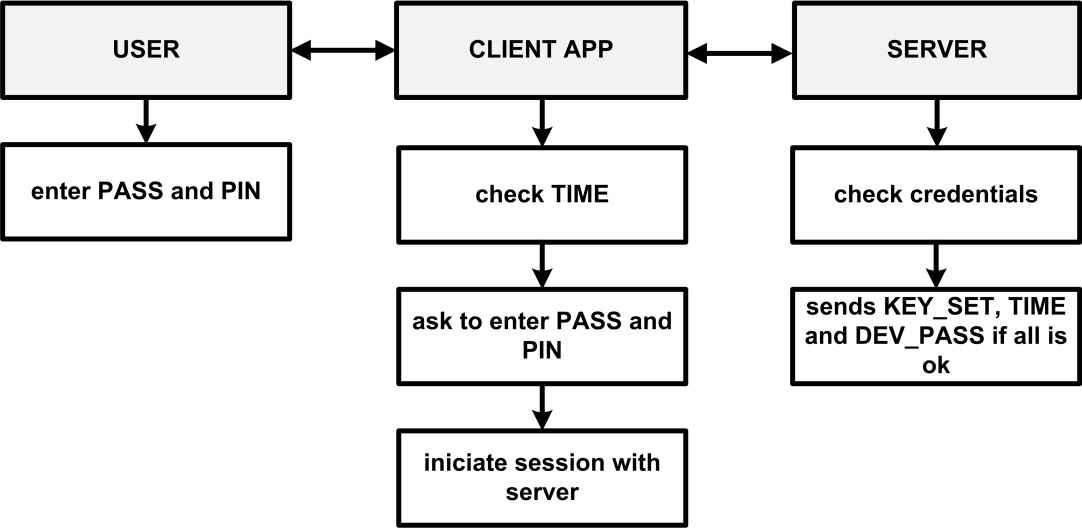
This section briefly outlines the main processes and functions in the online mode. The detailed description of this mode will be subject to be published in another paper. The connection to the network allows verifying the user credentials with the help of the protected cloud server. The communication with the cloud is secured by means of the encrypted key exchange protocol [20], [21] and serves to receive the new set of keys, files and send the collected log information. The client app operations in the online mode are as represented in figure 3.3.



*Figure 3.3*: Online mode operations

The communication with server is necessary once the key is expired. This allows to optimize the load on the network once the traffic is reduced to the discrete sessions. In such environment it is important to keep track of the time synchronization both on server and on client in parallel in order to avoid malicious user behavior messing up with time attacks.

The operation of the online mode starts from the client app sending the following token to server by means of EKE: DEV\_PASS+TIME+PASS+PIN in order to validate his identity and the identity of the user to the server. This value serves as a proof that the device and the user are the ones with which server communicated previously. After performing the verification the server sends the new DEV\_PASS (calculated) and TIME to the mobile app by means of established EKE session. After that server can send the new encrypted key set and encrypted files to the client app in the clear. The online workflow diagram is presented in Figure 3.4.



*Figure 3.4:* Online workflow diagram.

This model proposes to use light-weighted EKE for the fast communication with server communication like J-PAKE [20, 21] or SIS-based EKE [22]. SIS is a public key encryption that avoids generating the big primes, thus it can be used for the secure key exchange [22], [23]. EKE is performed as soon as the key is expired and the client asks the user to synchronize with the server. The security system based on the efficient SIS-EKE protocol is a subject of our next paper. Note that the EKE is not required to send the encrypted files or statistics.The current implementation of mobile app uses J-PAKE protocol. It is not the subject of this paper to describe the J-PAKE protocol and analysis in detail as it is analyzed in [20, 21, 24].

4. **Offline mode: our proposed model and solution**

This section proposes an open model of the mobile device protection mechanisms in which the security is supported both in online and offline modes. Currently, the systems of mobile device protection, to the best of the authors’ knowledge, follow the model where the protected mobile client can operate only when it is connected to the cloud, which is not always convenient for the end-user. The basic principles of the mobile device protection herein proposed are:

1. Optimized communication with the cloud when the device does not need to be constantly connected to the server due to the resource constraint and necessity to secure this communication;
2. Implementation of reliable cryptography standards, i.e. the algorithms AES, ABE and SSS are approbated and well-known by industry and academia. The idea is not to invent obscure concepts or to invent new methods that should be evaluated before the proposed usage in the provided solution;
3. Optimized combination of the security mechanisms so that the mobile client does not need to perform complex computation like encryption and key generation due to its resource constraint;
4. Behavioral analysis of user's operations on mobile client, which can indicate anomalous or automated activities performed by attackers.

The most important security issues in the proposed model arise when the device goes to the offline mode and the user is still allowed to get the access to the protected SME documents. In this case, the server can neither monitor the user activity nor provide the protection methods. The security becomes the responsibility of the mobile client. Additionally, the maximum protection should be provided at the minimum resource cost.

In the online mode the mobile device uses the secure communication with the server in order to verify the validity of user’s credentials. On the contrary, the offline protection model should be approached independently. Thus, the proposal is that the authentication/authorization mechanisms in the offline mode should utilize the derived proof of the user identity. The requirements for the proof are as follows:

1. The proof is derived from the previous session, in order to verify that the user is still authorized;
2. The proof should not give access to user password, i.e. it should not be stored in the device.
3. The proof should be temporary and have an expiration period;
4. It should not be directly used in communication with the server, in order to prevent the malicious user from mimicking;
5. It should be resilient to offline dictionary attacks;
6. It has to stay effective both in the scenarios of the malicious outsider and leakage of information when the formerly authorized user leaves the group.

Such proof cannot be stored on the client device, as it is not possible to guarantee its protection in the offline mode. Therefore, the most effective way to secure the proof in the offline without performing complex computation is to share this proof between the client and the user in a protected manner. In this case, there is no need to store the function of the user password. The additional argument against the traditional password verification is the necessity to check the PIN, which is very small, so the construction of valid one-way function resilient to the offline dictionary attacks is a difficult task.

The proposed offline mode architecture is represented in Figure 4.1.

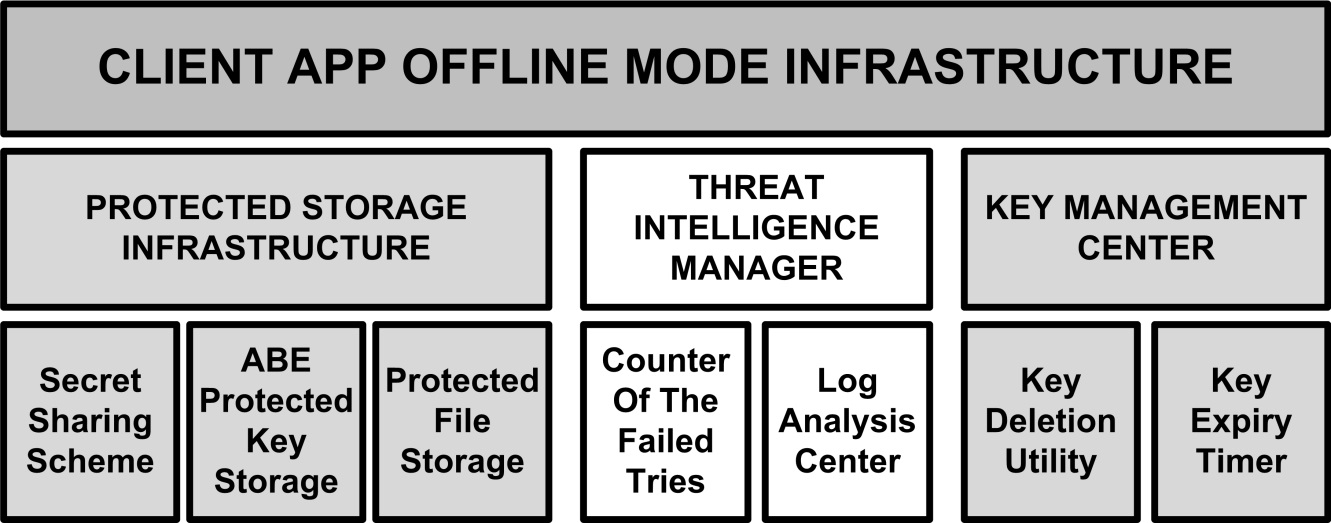


Figure 4.1: Offline security architecture

The infrastructure includes:

1. *The protected storage*: the storage is protected with the shared user key and contains the ABE keys giving access to the file keys which allow decrypting the stored files. The outline of the hybrid encryption scheme is presented in the Section 5.
2. *Threat Intelligence manager:* most attacks incur into significant variation on the legitimate behavior of information systems, or they adopt well-known patterns that can be easily detected by monitoring the system in the case of the offline mode. Signal processing techniques have been successfully applied to anomaly detection [25, 26] and have become a solution to a problem of improving detection accuracy, adaptability and computational cost for application on resource-constrained scenarios. Therefore, signal processing can be applied in offline mobile client security, for evaluating anomalies on user's behavior according to the scenarios in Section 6. Moreover, Model Order Selection (MOS), which is an effective signal processing technique to separate noise components from the principal components, can be applied into anomaly and attack detection [12], to identify and separate malicious behaviors from the legitimate ones.
3. *Key Management center:* it includes the functions for maintaining the key expiry period and deleting the expired keys.

**4.1. Offline mode workflow**

The user performs the following operations in the offline mode:

1. Enters the PASS and the 4-digit PIN;

2. Views the list of authorized files kept on the mobile storage;

3. Open the protected encrypted files;

4. Modify the protected files and save them in an unprotected storage.

The client app performs all the cryptographic calculations in the shadow. These calculations include the key storage, key restoring, decryption and showing the decrypted files in the client area. Note, that the client app does not check the password or PIN validity as it does not hold the verification proof for the above. Also, the app keeps track of the user activities and the key expiry. The complete list of the app activities is presented in the Figure 4.2.

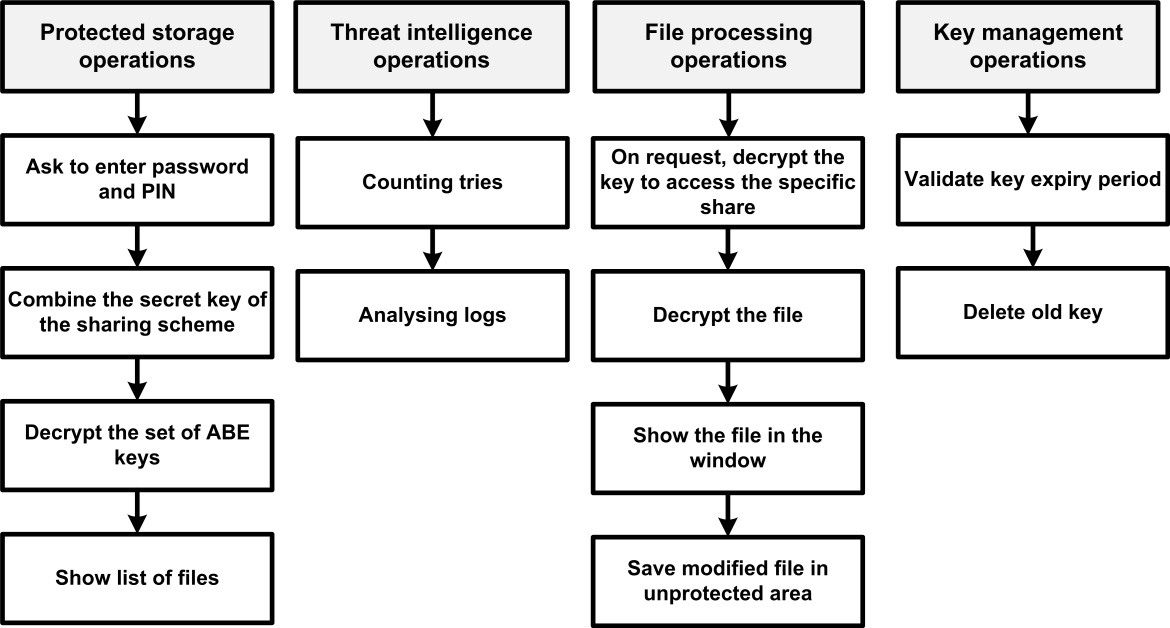


Figure 4.2: Offline mode operations chart

To finalize the description of the offline mode, Figure 4.3 shows the complete workflow of the proposed mobile application in the offline-mode.

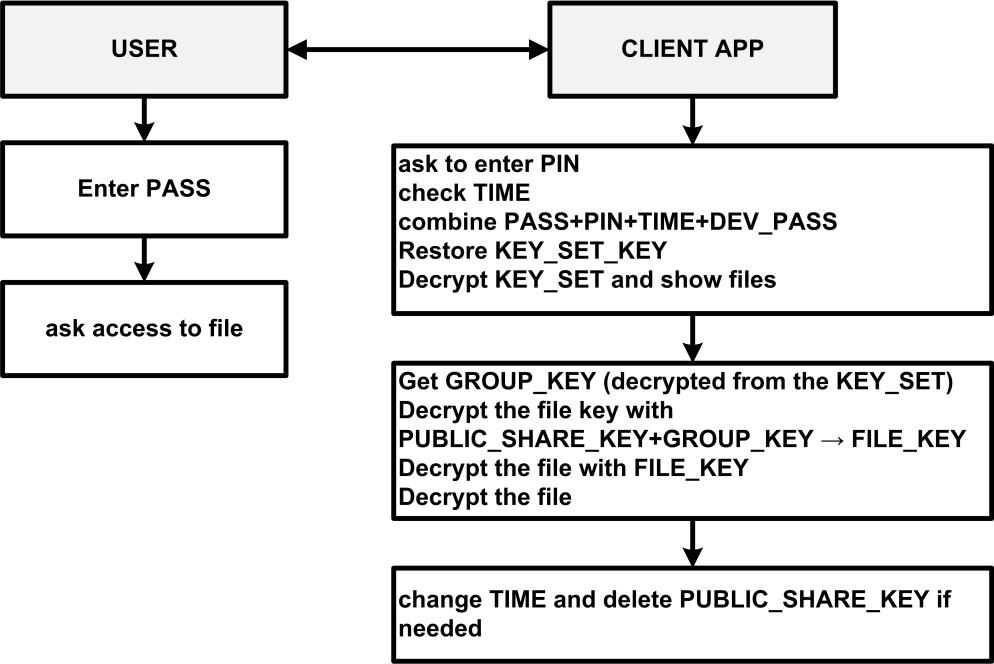


Figure 4.3: Offline mode workflow

The core cryptographic module of the implemented solution is based on the combination of AES-ABE-SSS methods, described in the Section 5. The key feature of the offline security is that the client app does not actually store any part of the user password to be verified. The client app combines its own key with the user share (PIN and password-derived) in order to restore the initial KEY\_SET\_KEY. If the user provides the wrong share the client will not be able to recognize it, but will decrypt the files incorrectly. Additionally, the password entering is tracked and too many tries in a short time are considered a threat.

**5. The algorithms, key usage and data protection methods.**

In the proposed approach, the kernel encryption scheme in the mobile device is a combination of several methods of security. The files are encrypted with 128/256-bit AES, while the permanent file keys are encrypted with the attribute-based encryption. The set of expiring ABE keys corresponding to the set of files accessible by user in encrypted with a single expiring AES key (KEY\_SET\_KEY). This key is expiring and is splited by the server into 4 parts (2 are stored on the device and 2 belong to the user) by the method of secure secret sharing. The encryption workflow is outlined in the Figure 5.1.

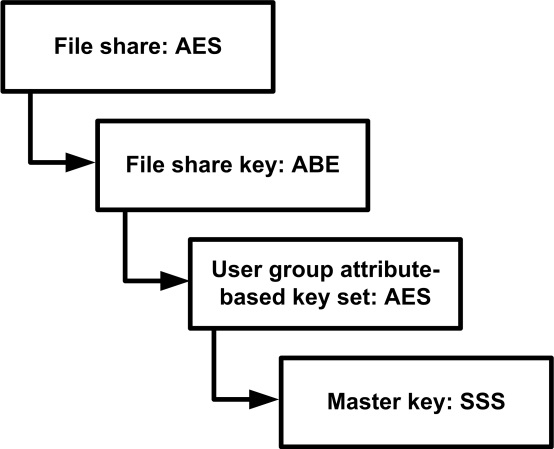


Figure 5.1: Encryption workflow

The key hierarchy can be explained by Figure 5.2 where the key represented in figure 5.1 serves for decrypting the key for each level represented in Figure 5.2.

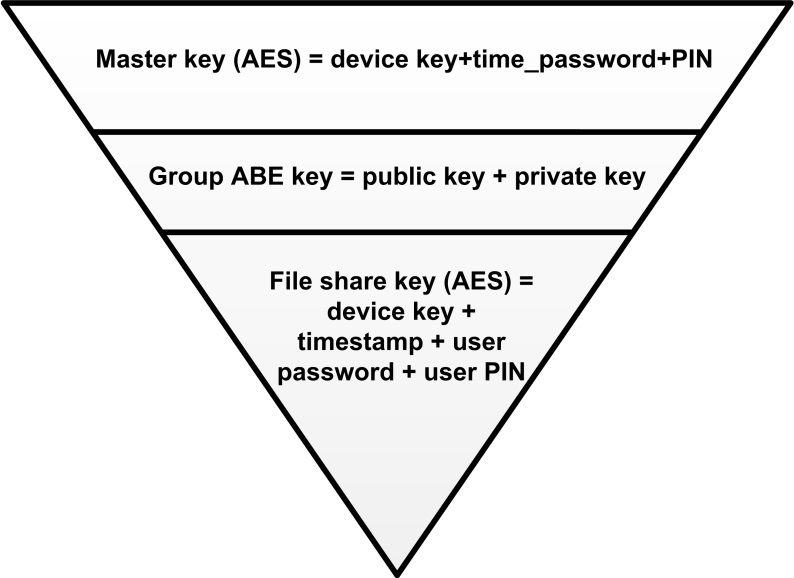


Figure 5.2: Encryption key hierarchy

This proposed encryption scheme is called as a hybrid encryption once it uses a combination of different encryption methods in order to support all requirements to the encryption scheme used in the proposed infrastructure. Then it has:

1. AES provides a fast encryption of the data files;
2. ABE permits to support the authorization policy on the encryption level;

3) SSS allows protecting the key storage by avoiding the necessity to store the key proof.

The following subsections 5.1.-5.3. provide a detailed description of the building blocks of the hybrid encryption.

**5.1. AES file encryption**

The protected files are encrypted on the server side with the secure 128/256 bit AES encryption, which is currently an industrial standard. Nevertheless, domain administrator can choose to encrypt the file storage with Blowfish or Serpent which also provide a high level of safety [27]. The symmetric encryption on a server side is performed in two steps. First there is a single AES key for preserving the server storage. Second, when the encrypted file is sent from the server to the mobile app or desktop device another unique AES key is generated. This key serves for the encryption on the client side. In other words, along with the encrypted file the user gets his own unique AES key for decrypting the files received. This key is protected by attribute-based public key encryption (ABE). The scheme of the server-side AES encryption is represented in Figure 5.3.

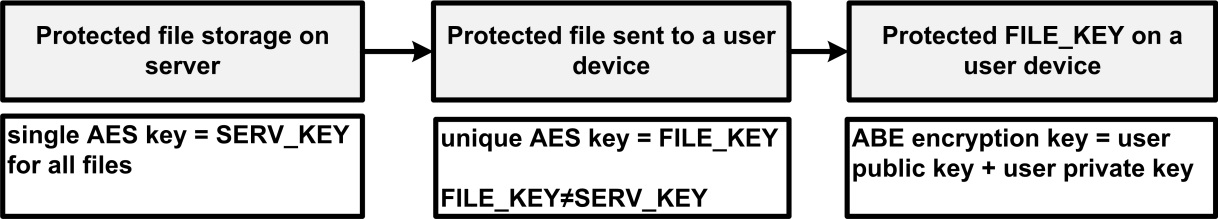


Figure 5.3: Server-side encryption

Usually, The AES encryption key is used as a session key. In our setting, it is not desirable to re-encrypt the files stored in the device unless there is a certain specific condition (for example, the user leaves the domain or a specific group in the domain (the group of users in the domain contains the users with the equal access rights, i.e. the same set of accessible file shares) and the file should not be accessible for this user anymore). This is why AES key in the presented notation is permanent and defined as FILE\_KEY. The randomly generated FILE\_KEY is unique for each file or set of files stored on a client.

**5.2. ABE encryption to protect the FILE\_KEY**

With each user session, the permanent FILE\_KEY (unique AES key) is re-encrypted. The set of FILE\_KEYs is protected with the corresponding ABE keys. The unique ABE model that we propose supports the attribute policy based on user groups and on file shares, i.e. the key attributes correspond both to the groups and the file shares.

The model supports the simple selective ABE scheme [28], [29]. The selective scheme for attribute-based encryption is as follows: if at least one attribute in the set{ti}U is equal to the attribute in the set{ti}M, the corresponding user U can decrypt the text M. In other words, as soon as user and share have one attribute in common – the user can get access to the share. The components of the ABE encryption are:

1. **Master-key (MK)** which is kept safely on server and accessible only for the domain administrator



The values tiare randomly selected from the huge group Zp. They are the private keys corresponding to the group attributes. Note, that this is different from the usual PK encryption: the private keys are controlled by the admin and not by the users.

1. **Public key (PK)** depends on the master key values and is kept in the clear allowing users to access the information:

Here e(g,g) is the bilinear pairing.

1. Secret user **KEY\_SET** depends on his attribute set. Here each Di (**GROUP\_KEY**) serves for decryption of the data of a single group of users, for example, related to some project:



1. **Encrypted text** M, in our context, M = FILE\_KEY, or the permanent AES symmetric key, which allows to avoid the file re-encryption.
2. **Encryption** procedure is multiplication. The set of the public keys Ei (**PUBLIC\_SHARE\_KEY**) corresponding to the set of groups able to access the text is kept along with the encrypted text E:
3. **Decryption** is division:



In order to perform this operation the user needs the pair of private key Di and public key Ei corresponding to the attribute ti and :



The result of decryption is the FILE\_KEY - the symmetric AES key that permits to decrypt the contents of protected file.

5.3. **Secret sharing scheme to protect the key storage**

The attribute-based private keys Di should be protected while being stored in the device memory. Therefore, server encrypts the set of Di with a single AES key before sending it to the user device. This AES key (master key) is denoted by the value KEY\_SET\_KEY in our notation. KEY\_SET\_KEY is a secret value and it is split by the secure method of polynomial modular secret sharing [30], [31] into the set of 4 shares:

KEY\_SET\_KEY = PASS+PIN+TIME+DEV\_PASS.

As the underlying sharing scheme is perfect [31], the adversary cannot get any information of the KEY\_SET\_KEY unless he is possession of all 4 key parts. Here the values PASS and PIN are predefined similar to the construction in [30].

The proposed authentication system is based on the shared storing of the user key. Also, the device acts as a dealer in the SSS. Using the SSS ensures that the key can only be accessed by an authenticated user. The participants of the (2, 2)-threshold SSS are the user and device. The user share  is computed based on the PIN -and the PASS entered by the user. Additionally, the current time value TIME is used in the calculation of s1(x).



Let  and ,



where – is a one-way function that transforms the data into the string of the desired length:





According to the CRT:



Thus calculated DEV\_PASS is written to the permanent device memory. The user share is not saved. Otherwise, it would allow an attacker to locally validate the restored private key.

5.4. **MOS for threat intelligence**

In the context of anomaly-based schemes for attack detection, the proposed behavioral analysis approach applies signal processing techniques, such as Principal Component Analysis and Model Order Selection schemes [12], for automatic identification of attacks or malicious behaviors.

Model Order Selection is an effective signal processing technique for several applications, allowing separating the only noise components from the principal components applying a rank reduction of the data. Applying MOS to the analysis of user operations can be effective in order to reveal the occurrence of malicious behavior during an offline session. MOS for threat intelligence requires that the target features, which can be user operations, be modeled as a matrix composed by the number of occurrences by location and time. For example, one element can represent the number of file readings on folder during the minute .



The MOS schemes can rely on correlation or covariance analysis, where the correlation analysis identifies anomalies on sparse or subtle number of file operations, while the MOS based on covariance analysis indicates abnormities caused by large amounts of operations during a period.

Classical approaches to model order selection require the computation of the sample covariance matrix and of its eigenvalues, obtained from the measurement matrix of the zero mean samples given by



The set of obtained vectors composes the zero mean matrix , then the covariance matrix can be calculated as follows



For MOS based on correlation analysis, instead of calculate the zero mean samples, to obtain the correlation matrix , it is required, for each variable, to make the standard deviation unitary as follows



The set of vectors composes the matrix , then the correlation matrix can be calculated via   
. Once the or have been obtained for MOS in order to anomaly detection, for the sake of simplicity, we refer to and as a matrix . Therefore, the next step of the algorithm is the eigenvalue decomposition (EVD), calculated according to , in order to obtain the vector of eigenvalues , as following:



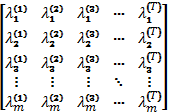
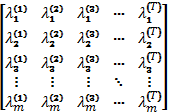
The eigenvalues should be sorted in descending order, as defined by , to make possible the selection of the first eigenvalue in the obtained sequence, represented by , which is the largest eigenvalue of the data evaluated for attack detection.



The process of obtaining the and the matrix , finding the largest eigenvalue for each -th time frame, should be repeated until t = T, in order to obtain the largest eigenvalue of all time frames, as presented by



=



Since , then the first line of the matrix contains the largest eigenvalues of each -th time frame, which is the expected input for MOS schemes and can be expressed as



Once obtained the largest eigenvalues of each q-th time frame, it is possible to apply a selected MOS scheme to estimate the model order d̂, which is the estimated number of time frames with malicious behavior. Therefore, is used as input parameter for MOS schemes, according to the equation .



Note that some MOS schemes may also require the amount of time that compose a time frame, such as . For more information about MOS schemes, interested readers are referred to [32].



1. **Results and analysis**

**6.1. Security analysis**

The security analysis of the proposed model was performed both from the point of implemented cryptographic mechanisms and from the user behavioral analysis. Two common attack scenarios were analyzed. First, the malicious outsider trying to infect or steal the important files. Second, the malicious expired user trying to steal the important files.

**6.1.1. Adversary model**

The proposal is to use a restricted adversary model. In other words, the actions of the adversary are predicted within the outlines model and scenarios. For example, it is not possible to guarantee the security of the decrypted document once the user copies its content to another file. In the analyzed scenarios, it is assumed that the adversary can:

1. Steal the user keys one-by-one while the app performs decryption, if he is outsider;
2. Use expired key to decrypt the files, if he is insider;
3. Does not have possibility to perform offline dictionary attack and cryptanalysis, i.e. does not have the possibility to verify user credentials apart from entering them in the client app;
4. Steal the encrypted files but not the decrypted ones.

**6.1.2. Analysis of security: results**

*Offline mode*

In the offline mode the client does not store the user password (see section 3 for details), i.e. no information about the password leaks, and therefore there is no possibility for the malicious user to check if the password he is trying to enter is correct or not. The only possible scenario for the information leakage in the case of the malicious outsider is if:

* 1. The hacker steals all the parts of KEY\_SET\_KEY;
  2. The hacker steals the KEY\_SET;
  3. The hacker try the brute force offline dictionary attack on all the previous values, they have to belong to ONE TIME SESSION (the values of ”a” and ”b” belong to one period of time i.e. TIME, DEV\_PASS, KEY\_SET);
  4. Steal the permanent FILE\_KEY (this is protected by the KEY\_SET);
  5. Steal the file and try to decrypt it with offline dictionary attacks.

Still, the hacker has to get 4 values: TIME, DEV\_PASS, KEY\_SET, FILE\_KEY - from one session. At the same time he should try the offline dictionary attack on PIN+PASS. Moreover, the 4 values provide access only to 1 single file. So practically, it is very difficult to perform such attack due to the key expiration period. In accordance with the conditions 1) and 3) of our adversary model, the above actions are not possible for such adversary in the conditions presented.

The temporary nature of all parameters obliges the user to connect to server when necessary and prevents the malicious actions from the user side. The only possible scenario when malicious or expired user wishes to prolong his old credentials is:

* 1. He has to steal the DEV\_PASS and TIME synchronized with his credentials;
  2. He has to be able to combine;
  3. He has to steal the KEY\_SET synchronized with his credentials;
  4. He has to steal the protected FILE\_KEY (also synchronized) and encrypted file;
  5. He has to do everything without the client (because the client checks the TIME and renews the PUBLIC\_SHARE\_KEY).

Basically, for a malicious user there is practically no way to use the client with the old keys. The fact that the client does not contain any data to be checked or validated does not prevent the user from seeing the contents of decrypted files, but with the wrong password the decrypted files will be different from the original ones wrong.

The client still has to count the tries (to prevent the hacker to perform brute force attack) within one session. In accordance with the conditions 2) and 4) of the adversary model, the above actions are not possible for such adversary.

*Online mode*

In the online mode the protection of the mobile system is backed up by the communication with server. The key expiry period and the usage of J-PAKE [20, 21] guarantees that the sensitive data sent over the network cannot leak. The detailed security analysis of J-PAKE is presented in [24]. The data sent in the clear i.e. encrypted files and the public keys of ABE does not need to be protected additionally as the AES and ABE guarantee its security.

**6.1 3. Common threat scenarios.**

Here it is presented the detailed description of the common scenarios in which the log and behavioral analysis is provided. The behavioral analysis can help to keep the user or administrator informed of the threat and take actions, as well as can be useful in order to implement threat preventions or reactive actions to avoid threat propagation.

**6.1 3.1 An attacker uses a valid password to perform operations on a bulk of files;**

The session time defines the period when operations can be performed until the next session renewing. During this period, it is still necessary to identify attacks and malicious behavior on file operations, in order to avoid fast attacks to perform unauthorized access to information or data modification. Some attacks present behavioral patterns based on abrupt number of operations, such as the ransomware attack, which is a growing attack [33] that blocks the access to valuable resources and requires a payment in order to unblock the content. The access to the resources can be blocked by the attacker through some techniques, when the content is encrypted by the attacker, the ransomware attack can be called cryptoransomware [34].

MOS schemes based on covariance analysis are effective to reveal abrupt changing on behaviors over time [12], making possible to identify intense malicious behaviors on offline mode of mobile clients, such in case of ransomware attack or bulk access to sensitive data.

The large number of operations over time is a well-known pattern of some attacks, due to the efforts on security measures to make the attacks infeasible over time. In this context, the operations can also be evaluated in contrast to the estimated required time for operations done by legitimate behaviors, such as the evaluation of the mean time between operations, highlighting the occurrence of infeasible behaviors in comparison to legitimate user activities.

Sparse or subtle file operations, with low number of operations distributed over different files or directories, during short period of time can indicate anomalies in contrast to the required time for legitimate directory navigation. MOS and correlation analysis can be suitable if applied to evaluate the time and location of operations, in order to identify unreachable navigation, if compared to legitimate navigation

The MOS based on covariance analysis indicates abnormities caused by large amounts of operations during a period. The eigenvalue analysis based on the correlation matrix highlights massive or concentrated operations over time or folder location, which is evaluated by MOS schemes in order to identify the number of malicious behaviors during the evaluated time.

This threat scenario, where an attacker uses a valid password and session to perform operations on a bulk of files, can have its steps described as:

1. The hacker has access to the mobile client and is able to perform operations;
2. The session time is valid;
3. The hacker tries to perform legitimate operations, such as file decryption, encryption, reading, writing or directory navigations;
4. The client incrementally append each operation attempt time into the logging;
5. The MOS module evaluates the logging of legitimate operations, applying correlation analysis to identify anomalies on sparse or subtle number of file operations, highlighting the occurrence of infeasible behaviors in comparison to legitimate user activities;
6. The MOS module evaluates the logging of legitimate operations, applying covariance analysis to identify abnormities caused by massive operations during the session time;

**6.1 3.2 Usage of expired password to perform non-authorized operations**

In the offline mode, the session time is used to restrict the operations during a specified period, although it is possible to manipulate the current time in mobile clients, to emulate a period in which the session was valid. The log analysis by MOS can deal with this kind of threat, through the incremental logging of the time when each operation was performed, followed by the behavioral evaluation of operations over time.

The incremental logging assumes that new logged operations shall have equal or bigger time than the last logged operation, the violation of this rule means that the system is out of sync and indicates a malicious behavior. Additionally, a large amount or sparse operation performed at the same time, or during a short period, can indicate the use of backtrack techniques to maintain a valid session during necessary time to perform an attack. Massive, subtle or sparse malicious operation performed during a valid session time can be identified by MOS schemes based on covariance and correlation analysis.

Applying MOS to the analysis of the time between user operations can be effective in order to reveal the occurrence of malicious behavior during an offline session. The MOS based on correlation analysis identifies anomalies on sparse or subtle number of file operations, since the eigenvalue analysis based on the correlation matrix highlights the unexpected number of sparse (such as file operations on diverse folders) or subtle operations. Consequently, the result of the eigenvalue analysis is applied to MOS schemes, in order to identify the occurrence of malicious behaviors during the valid session.

The MOS based on covariance analysis indicates abnormities caused by large amounts of operations during a period. The eigenvalue analysis based on correlation matrix highlights massive or concentrated operations over time or location, which is evaluated by MOS schemes in order to identify the number of malicious behaviors during the evaluated time.

This threat scenario, where the attacker uses expired password to perform non-authorized operations, can have its steps described as:

1. The hacker steals the operating system;
2. The hacker modifies the time of the operating system to a period when the session was valid;
3. The hacker has access to the mobile client and is able to perform operations;
4. The hacker tries to perform legitimate operations, such as file decryption, encryption, reading, writing or directory navigations;
5. The client incrementally append each operation attempt time into the logging;
6. The client verifies if one logged time is older than the last operation time. If it is true, the MOS module classifies the evaluated operation as malicious;
7. The MOS module evaluates the logging of legitimate operations, applying correlation analysis to identify anomalies on sparse or subtle number of file operations;
8. The MOS module evaluates the logging of legitimate operations, applying covariance analysis to identify abnormities caused by massive operations during the session time;

**6.1 3.3 Data Modeling for Behavioral Analysis**

We use the MOS schemes in order to identify anomalous behavior that can indicate an attack and be used to prevent or avoid attack propagation. Therefore, it is necessary to analyze the data that can be collected from user operations on mobile client, to identify features that can be modeled and submitted to MOS schemes, according to described in Section 5.4.

The selected features shall be modeled as matrices which represents a signal superposition containing noise, legitimate and malicious behavior [12], grouped into time frames , where M defines the decomposition of a selected feature, N defines the time decomposition and represents the number of occurrences of the feature during the time .



On offline mode, the user is still allowed to get access to operations that do not require communication with the server side. These operations and their selected features shall be incrementally logged by the mobile client, in order to be evaluated to identify malicious behaviors. This work proposes to evaluate the following features:

* File Access (Time and File System Location);
* File Update (Time and File System Location);
* File Download (Start Time, End Time and File System Location);
* File Upload (Start Time, End Time and File System Location);

These features provide the necessary information to apply MOS schemes for evaluating malicious behaviors according to the following scenarios and steps:

1. The hacker accesses the mobile client during a valid session and read/write few files at several folders:
   1. The hacker steals the device;
   2. The hacker changes the device to offline mode;
   3. The hacker accesses the mobile client;
   4. The current session is valid;
   5. The hacker read/writes few files at several folders;
   6. MOS based on correlation analysis is applied to the matrix of number of file operations, grouped by the file system location over time;
   7. MOS indicates the number of time frames with anomalous behavior;
2. The hacker accesses the mobile client during a valid session and read/write several files at few folders:
   1. The hacker steals the device;
   2. The hacker changes the device to offline mode;
   3. The hacker accesses the mobile client;
   4. The current session is valid;
   5. The hacker read/writes several files at few folders
   6. MOS based on covariance analysis is applied to the matrix of number of file operations, grouped by the file system location over time
   7. MOS indicates the number of time frames with anomalous behavior;
3. The hacker emulates a valid session and read/write few files at several folders:
   1. The hacker steals the device;
   2. The hacker changes the device to offline mode;
   3. The current session is expired;
   4. The hacker changes the device time to a valid period for the current session;
   5. The hacker accesses the mobile client;
   6. The current session is valid;
   7. The hacker perform sparse operations during the valid period;
   8. MOS based on correlation analysis is applied to the matrix of number of file operations, grouped by the file system location over time;
   9. MOS indicates the number of time frames with anomalous behavior;
4. The hacker emulates a valid session and read/write several files at few folders:
   1. The hacker steals the device;
   2. The hacker changes the device to offline mode;
   3. The current session is expired;
   4. The hacker changes the device time to a valid period for the current session;
   5. The hacker accesses the mobile client;
   6. The current session is valid;
   7. The hacker perform bulk operations during the valid session;

MOS based on covariance analysis is applied to the matrix of number of file operations, grouped by the file system location over time;

* 1. MOS indicates the number of time frames with anomalous behavior;

**6.2. Complexity analysis**

The complexity analysis depends on the operation that the user and the mobile app perform in order to keep the device protected. In the offline mode the client app performs the following actions:

1. Combine the PASS+PIN+TIME+DEV\_PASS=KEY\_SET\_KEY → SSS secret restoring;
2. Decrypt the KEY\_SET with the KEY\_SET\_KEY → symmetric 128/256 AES decryption;
3. Select the SHARE\_KEY from the KEY\_SET → no calculation;
4. Decrypt the FILE\_KEY with the SHARE\_KEY → ABE decryption;
5. Decrypt the file with the FILE\_KEY →symmetric 128/256 AES d→ecryption;
6. Modify the TIME periodically → timer;
7. Count the tries within the TIME →count;
8. Modify or delete PUBLIC\_SHARE\_KEY→ no calculation;
9. Analyzes the log data and deletes the files when necessary -------------------- TODO

It is easy to check that in the offline mode the client app does not perform complex calculations and does not use the resources extensively due to the fact that the initial key is shared and the client app only performs decryption, which is not a time-consuming operation. Thus, the proposal supports the concept of the light-weighted client, i.e. the most consuming operations are ABE and AES decryption.

Similarly, in the online mode the client app does not perform resource-consuming operations apart from J-PAKE construction which is used to renew the user/app master key (DEV\_PASS):

1. Generates keys for J-PAKE → this can be a resource-consuming operation. In the future we suggest to replace J-PAKE with SIS-based PKE [22, 23];
2. Sends and receives data in the clear;
3. Performs operations 1)-7)
4. Analyses and sends the log data ------- TODO.

The proposed concept of mobile client security has been implemented in the Storgrid protected cloud environment (www.storgrid.com). Therefore, the approach is correlated with the practical usability requirements: the corporate user continues to use the mobile storage app in offline and does not need to reload the files every time the key is renewed. This methodology can be used in other mobile apps. The common advantage is that the mobile client performs the operations both in the offline and online mode and uses the key expiry and ABE to protect the privacy of the corporate data. The table below presents the results of mobile app security workflow testing.

Table 1. Speed of decryption on a client device.

Table 2. Speed of key renewal operation

Table 3. Speed of analytics (?)

We may conclude that the proposed client is not overloaded with calculations due to the carefully selected mathematical operations. It can be successfully used and provides acceptable level of security.

1. **Conclusion and future work**
2. **Our approach is a practical application to be used in the corporate mobile environment. It is implemented as afully working mobile app. WE suggest that it can be used for any type of enterprise. Also, part of concept is sed in the big data apps. We plan to enhance the analytics and implement the effective protocols.**

**References**

[1] Cloud Security Alliance. Official page. Available via <https://cloudsecurityalliance.org/>. Cited 15 Jan 2016

# [2] Higashi M. (2015) [Cloud Data Security and EU Data Privacy Rules Compliance with Encryption and Tokenization](http://www.ciphercloud.com/blog/cloud-data-security-and-eu-data-privacy-rules-compliance-with-encryption-and-tokenization/" \o "Permanent Link: Cloud Data Security and EU Data Privacy Rules Compliance with Encryption and Tokenization) In: Cloud Security, Compliance. Available via http://www.ciphercloud.com/blog/cloud-data-security-and-eu-data-privacy-rules-compliance-with-encryption-and-tokenization/. Cited 15 Jan 2016

[3] Gartner, Inc. (2015) [Key](http://www.ciphercloud.com/blog/cloud-data-security-and-eu-data-privacy-rules-compliance-with-encryption-and-tokenization/" \o "Permanent Link: Cloud Data Security and EU Data Privacy Rules Compliance with Encryption and Tokenization) Challenges in Cloud Computing In: Cloud Computing. Available via <http://www.gartner.com/technology/topics/cloud-computing.jsp>. Cited 15 Jan 2016

[4] Skyhigh research and reports (2015) What is Cloud Access Security Broker. In: Skyhigh Cloud University. Available via <https://www.skyhighnetworks.com/cloud-university/what-is-cloud-access-security-broker/>. Cited 01 Feb 2016.

# [5] Lawson C.,  MacDonald N.,  Lowans B. (2015) Market Guide for Cloud Access Security Brokers. In Gartner research. Available via <http://www.gartner.com/technology/reprints.do?id=1-2RUEH70&ct=151110&st=sb>. Cited 01 Feb 2016.

[6] Campbell M. (2015) Cloud Data Encryption is Easy. In Cloud Cyphercloud blog. Available via <http://www.ciphercloud.com/blog/cloud-data-encryption-easy/.Cited> 01 Feb 2016.

[7] Yovel **Y.** (2014) 4 Essential Ways to Protect My Mobile Apps. In Security Intelligence e-magazine. Available via <https://securityintelligence.com/how-to-protect-mobile-apps-essentials/>. 02 Feb 2016

[8] Abdul Nasir Khan, M. L. Mat Kiah, Mazhar Ali, Shahaboddin Shamshirband, Atta ur Rehman Khan. A Cloud-Manager-Based Re-Encryption Scheme for Mobile Users in Cloud Environment: a Hybrid Approach. Journal of Grid Computing. 13 (4), 651-675 (2015).

[9] Abdul Nasir Khan, M. L. Mat Kiah, Mazhar Ali, Sajjad A. Madani, Atta ur Rehman Khan, Shahaboddin Shamshirband. BSS: block-based sharing scheme for secure data storage services in mobile cloud environment. Journal of Supercomputing. 70 (2), 946-976 (2014).

[10] Galibus, T., Matveev, G. Generalized Mignotte Sequences in Polynomial Rings. ENTCS. 186, 39-45 (2007).

[11] Goyal, V., Pandey, O., Sahai, A. & Waters B., Attribute-based encryption for fine-grained access control of encrypted data. *Proceedings of the 13th ACM conference on Computer and communications security*, 89–98 (2007).

[12] Tenorio, D.F., da Costa, J.P.C., de Sousa Jr, R.: Greatest eigenvalue time vector approach for blind detection of malicious track. Proceedings of the International Conference on Forensic Computer Science (ICoFCS) 2013. Brasília, 14-16 August 2013. P. 46-51.

[13] Khan, A.R., Othman, M., Madani, S.A., Khan, S.U.: A survey of mobile cloud computing application models.Communications Surveys & Tutorials, IEEE. **16**, 393–413 (2014).

[14] Khan, A.N., Mat Kiah, M., Khan, S.U., Madani, S.A: Towards secure mobile cloud computing: a survey. Futur.Gener. Comput. Syst. **29**, 1278–1299 (2013).

[15] Zhao, G., Rong, C., Li, J., Zhang, F., Tang, Y. Trusted data sharing over untrusted cloud storage providers. *IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom’10)*, Washington, DC, USA, 97-103 (2010).

[16] Yang, J., Wang, H., Wang, J., Tan, C., Yu, D. Provable data possession of resource-constrained mobile devices in cloud computing. Journal of Networks. **6**, 1033–1040 (2011).

[17] Itani, W., Kayssi, A., Chehab, A.: Energy-efficient incremental integrity for securing storage in mobile cloud computing. *International Conference on Energy Aware Computing (ICEAC ’10)* Cairo, Egypt, 1-2 (2010).

[18] Ren, W., Yu, L., Gao, R., Xiong, F. Lightweight and compromise resilient storage outsourcing with distributed secure accessibility in mobile cloud computing. Tsinghua Science & Technology. **16**, 520–528 (2011).

[19] Van Lelyveld, A. (2013) SAP Mobile Platform Secure Mobile with Mocana. In SMP Enterprise Grade Mobility – Webinar Series. Available via: <http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/8063ed15-0713-3110-c584-e75ac0395b20?QuickLink=index&overridelayout=true&58725087881424>. Cited 15 Feb 2016.

[20] Hao F., Ryan, P. J-PAKE: Authenticated Key Exchange without PKI. LNCS [Transactions on Computational Science XI](http://link.springer.com/book/10.1007/978-3-642-17697-5). 6480, 192-206 (2010).

[21] Hao F., Ryan, P. Password Authenticated Key Exchange by Juggling. LNCS [Security Protocols XVI](http://link.springer.com/book/10.1007/978-3-642-22137-8). 6615, 159-171 (2008).

[22] Bogos, S., Boureanu, I., Vaudenay, S. Primeless Factoring-Based Cryptography: Solving the Complexity Bottleneck of Public-Key Generation. LNCS [Applied Cryptography and Network Security](http://link.springer.com/book/10.1007/978-3-642-38980-1). 7954, 552-569 (2010) ––

[23] Bellovin, S. M., Merritt, M. Encrypted key exchange: Passwordbased protocols secure against dictionary attacks. *IEEE Symposium on Security and Privacy.* 72–84 (1992)

# [24] [Toorani, M](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Toorani,%20M..QT.&newsearch=true). Security analysis of J-PAKE. *Computers and Communication (ISCC), IEEE Symposium on (2014)* doi:[10.1109/ISCC.2014.6912576](http://dx.doi.org/10.1109/ISCC.2014.6912576).

[25] Lu, W., Ghorbani, A.: A. Network anomaly detection based on wavelet analysis. EURASIP J. Adv. Signal Process. 2009, Vol. 4(1416 ). P. 1-17.

[26] Huang, C.-T., Chang, R.K.C., Huang, P.: Signal processing applications in network intrusion detection systems. EURASIP J. Adv. Signal Process. 2009. Vol. 9(192). P. 1-4.

[27] NIST (2000) [*Report on the Development of the Advanced Encryption Standard (AES)*](http://csrc.nist.gov/archive/aes/round2/r2report.pdf). Available via PDF: http://csrc.nist.gov/archive/aes/round2/r2report.pdf

[28] Galibus, T., Vissia, H. Cloud Storage Security. *Proceedings of NSCE’2014, 24-25 December 2014, Hong Cong.* 123-127 (2015).

[29] Galibus, T. Access control for the cloud storage. *Proceeding of the 3rd Belarus-Korea Forum Science “Innovation, Production”, Minsk, 16-17 Oct. 2014*. 72-73 (2014).

[30] Galibus, T., Gafurov, S., Kaganovich, D., Vissia, H. Mobile security based on the secret sharing. The journal of Brest state technical university. 5, 33-36 (2015) (In Russian).

[31] Galibus, T., Matveev, G., Shenets, N. Some structural and security properties of the modular secret sharing. *Proceedings of IEEE SYNASC'2008*.197-200 (2009).

[32] da Costa, J.P.C.L., Thakre, A., Roemer, F., Haardt, M.: Comparison of model order selection techniques for high-resolution parameter estimation algorithms. In: Proc. 54th International Scientific Colloquium (IWK'09), Ilmenau, Germany. 2009.

[33] McAfee Labs Threats Report. McAfee. Part of Intel Security. August 2015. Available: http://www.mcafee.com/us/resources/reports/rp-quarterly-threats-aug-2015.pdf

[34] Mobile Cyber Threats. Kaspersky Lab & INTERPOL Joint Report. October, 2014. Available: <http://media.kaspersky.com/pdf/Kaspersky-Lab-KSN-Report-mobile-cyberthreats-web.pdf>