Here goes the title page:

**Mobile client security architecture: a practical approach**

**A practical approach to the mobile client security architecture**

(???)

**Authors affiliations:**

Those who participate now: Tatiana, Thiago, Edison

From our side I will include the guys from Strgrid when we start the implementation chapter.

(pls, add yours who is going to participate)

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

Abstract:

This paper presents a novel approach to the mobile client security. It includes the concept and definition of offline mobile security, i.e. the protection of data when the mobile client is not connected to the cloud. In order to preserve the security of data we use the cryptographic methods, such as AES encryption to protect the files, ABE encryption based both on user and share attributes to protect the user keys, SSS to share the sensitive key between the device and the user and MOSto provide the analysis of user behavior. We optimize the usage of the methods in order to save the energy of the mobile client and reduce the communication with the cloud.

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

Cloud computing is a new paradigm in the world of distributed networking and computation. The basic feature of the cloud environment is providing the elastic, on-demand and secure service for the end-users. While the first two requirements are rather well supported by the cloud platforms in use, the security is a major 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 a major issue.

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 the 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. The reason for this is that the organization of the 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 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 in the user side [6].

The additional security problems and requirements need to be considered when the mobile devices are actively used in the cloud environment [7]. Today the society lives in the BYOD (Bring-Your-Own-Device) world and the mobile devices pose a serious risk to the SME cloud platforms as the bottleneck of the information security system (ISS). While the enterprise web interface and the cloud environment can be protected by powerful third-party services, such as CASB and CAC, the mobile client 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 the battery power limitations [8]. Therefore, the encryption on a mobile device is not a good solution: the proposed schemes are computationally good, but lacks the security analysis in many cases [9]. The common practice is the shadow user activities monitoring [7]. However, one area of the mobile device usage stays unprotected in all the proposed scenarios.

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). However, what happens when the mobile client goes offline? And even worse - with some sensitive data on board? All powerful cloud-based tools cannot help and the mobile app has to secure itself with its own limited resources. Moreover, the critical point is the difference in strategy of online and offline protection. Due to its resource constraint, it is not possible to perform the extensive computation and encryption on the mobile device.

Observing the above described landscape, this paper considers the concept of the offline mobile client security. The proposal describes 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), 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. With this goal, the key expiration period is used and safely incorporated into the proposed system solution. Additionally, the behavioral analysis can indicate well known malicious behaviors, their variations, as well as novel attacks, that present low or high variance in comparison to legitimate user behaviors. The main target 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. The text mainly concentrates on the offline protection in the BYOD world. Section 3 gives the basic definitions and explanations of the used methods: SSS, ABE and MOS. Section 4 presents the complete proposed solution to the problem of offline mobile client security. Section 5 traces the workflow of the client activity and analyzes the common security breaches. As discussion about the security proofs for the proposed system is also provided in Section 5. Section 6 presents the practical implementation and analysis of complexity of the proposed solution, while Section 7 concludes the paper.

**Related Work**

The BYOD world requires more from the traditional data protection services compared to ordinary computing environment. Apart from authentication, DLP services and encryption (at rest, in transit, in use) it is necessary to provide additional contextual methods [REF]. The contextual methods increase the security of the app at a maximum level with a minimum resource demand. The traditional ones are:

1. Using geo location;
2. Expiration of an app;
3. Secured transfer between apps;
4. Restricted access to app;
5. Expiring pass/pin;
6. Counting of failed tries;
7. Offline protection approach;
8. Audit.

Therefore, it is possible to conclude that it is harder to protect data on the mobile device so the owner or the SME should take more care of protecting its data leaving the organization. According to this perspective, the data on a mobile device should be considered as one leaving the organization. The most sensitive and confidential data should not be allowed to be transferred to the mobile device at all. 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 [REFs]. 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 the new way of using data storage services. An analysis of the existing models of mobile cloud computing is presented in [8]. All the models and protection schemes concentrate on the encryption properties and either perform the computations on their own [12-15] or use the cloud provider to off-load the cryptographic operations [16-18]. It is natural that the mobile device cannot handle the operations securely without assistance of a cloud provider, due to the resource constraint and battery limitation.

When it is desired to make the device more independent and less dependent on the cloud provider (suppose, an application needs to run securely without connection to the network) it is possible to use only the schemes that function without putting load on a provider. All the currently known schemes of encryption, performing the computation, as it is discussed in [8], either use the a cloud provider [12], a third party trusted agent [13], a combination of both [14], or concentrate on computational complexity without taking care of user privacy and security[15].

In other words, the security schemes proposed so far, are not working offline. In many cases the industrial providers of the secured application the security functions API, that protects the mobile app like a wrap (like Mocona, operating along with SAP [16]), preferring to completely forbid the offline access to the protected app.

In some cases such access is still necessary. Due to such constraints as traffic load, travelling, ease of access and many more. So the question is how to protect the device/app, in an offline mode when the functions of data protection cannot be offloaded to a cloud or a trusted party. This problem was not approached neither in academia nor in the industry.

**Offline mode**

This paper 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 that is supported by the here proposed approach 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. 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 strategic organization 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 device communicate with the server in order to check the validity of user password. On the contrary, the offline protection should implement a different approach. In other words, the authentication/authorization mechanisms in the offline mode should utilize the specific 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 shoud 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 the off-line dictionary attack.
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.

If such proof can be constructed, then the offline mode can be secured. The most important requirement is that the password (or the proof of the password) cannot be stored on the client device, as it is not possible to guarantee that it is perfectly protected in this case. The client cannot be in possession of the user knowledge as these are the separate entities. Rather, this proof should be shared between the client and the user in a secure manner. This is the only way the client can support the security without performing complex computation or storing the function of the user password.

In the mobile client, the user keys are protected by the combination of user password and PIN as well as the ABE keys that have an expiration period. 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.

Most of the attacks incur into significant variation on the legitimate behavior of information systems, or adopts well-known patterns that can be easily detected by monitoring the system. Signal processing techniques have been successfully applied to anomaly detection [17,19] 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 following scenarios.

Moreover, Model Order Selectio, which is an efferctive signal processing technique to separate noise components from the principal components, can be applied into anomaly and attack detection [18], to identify and separate malicious behaviors from the legitimate ones.

1. **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 be used to 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 can indicate a malicious behavior. Additionally, a large amount or sparse operation performed at the same time can indicate the use of backtrack techniques to maintain a valid session during necessary time to perform an attack.

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, and the MOS based on covariance analysis indicates abnormities caused by large amounts of operations during a period.

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 information access or data modification.Some attacks present behavioral patterns based on abrupt number of operations, such as the ransomware attack, which is a growing attack [21] 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 throught some techniques, when the content is encrypted, the ransomware attack can be called cryptoransomware [20].

MOS schemes based on covariance analysis are effective to reveal abrupt changing on behaviors over time [18], 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, 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 algorithms and definitions.**

This proposal consists of a combination of secure and effective methods in order to protect the client as detailed explained in the following.

1. **AES encryption to protect the files.**

The protected files are encrypted on the server with the secure and well-known AES cypher, which is currently an industrial standard. Other options include Blowfish and Serpent. All these are block symmetric cyphers providing high level of safety. In other words without knowing the AES key it is impossible to decrypt the files.

The AES encryption key is normally used as a session key. It is not desirable to re-encrypt the files stored in the device unless there is a certain condition (user leaves the group and the file should not be accessible). That is why AES key in the presented notation is defined as FILE\_KEY. If it is not desirable that the user have access the file, the system kills the FILE\_KEY, so that user has to perform a hard cryptanalysis to reveal the file. The randomly generated FILE\_KEY is unique for each file share stored on a client. So, the user needs to perform the cryptanalysys for each file separately.

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

The permanent file key is re-encrypted with each session. The KEY\_SET is defined as the set of all FILE\_KEYs. The KEY\_SET\_KEY protects this KEY\_SET (a symmetrical AES key). The proposed model supports both the authorization based on user groups and on the shares, i.e. there is a separate KEY entity corresponding to each share in the group.

The selective scheme for attribute-based encryption is as follows.

If at least one attribute in the set{t\_i}\_U is equal to the attribute in the set{t\_i}\_M, the corresponding user U can decrypt the text M.

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 ti are 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:
2. **Encryption** of the text M (in this proposal the text is the **FILE\_KEY**, or the permanent AES symmetric key, the key is permanent in order to reduce the necessity to re-download the files on the device) 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:
3. **Decryption** is division:

In order to perform this operation the user needs Di corresponding to the secret attribute ti and :

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

3. **The KEY\_SET\_KEY is protected via the secret sharing.**

In other words, it is considered as a secret value and it is split (by the modular sharing) into the set of 4 shares:

PASS+PIN+TIME+DEV\_PASS (where the PASS and PIN are predefined, as in [brest] and TIME is modified ). In order to get the KEY\_SET\_KEY the hacker has to get all 4 parts (otherwise he gets no information of the secret due to the perfect nature of SSS).

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, device and the time mark. The user share  is computed based on the **-**and the PASS entered by the user. The TIME is the current time value. 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\_KEY is written to the permanent memory. The user share is not saved. Otherwise, it would allow an attacker to locally validate the restored private key.

4. **MOS**

In the context of anomaly-based schemes for attack detection, the proposed behavioral analysis approach applies signal processing techniques, such as Principal Component Analyis and Model Order Selection schemes [18], for automatic identification of attacks or malicious behaviors.

Model Order Selection is a effective signal processing technique for several applications, allowing to separate the only noise components from the principal components applying a rank reduction of the data. 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 anomalys 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, we refer to [22].

**Offline mode: proposed solution**

Figure XXXX shows the workflow of the proposed mobile application in the offline-mode:

The approach is based on the combination of AES-ABE-SSS methods. The key feature of this approach is that the client does not actually store any part of the user password to be verified. The client 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.

**Online mode**

**In the online mode the client acts differently. There is a support from server and the user credentials can be checked once the validity period of his keys expires. So the communication with server happens once the client discovers the time is expired.**

1. **Client checks the time counter and the keys validity.**
2. **If time is expired the client deletes the PUBLIC\_SHARE\_KEY of the user.**
3. **The user does not have access to the files anymore.**
4. **The client asks the user to connect to the server (or does it in the background)**
5. **User enters the PASS and PIN**
6. **The client uses the safe protocol for key exchange (for the moment, J-PAKE) to establish the new KEY\_SET and KEY\_SET\_KEY**
7. **After receiving the KEY\_SET and TIME+DEV\_PASS the client sets up the time counter.**
8. **From this moment the communication with the server is not necessary. User just needs to enter his PIN in order to decrypt the protected files.**

**It is important to keep track of time both on server and on client in parallel in order to avoid the malicious user behavior. In such concept of a synchronized communication with server, when the client does not need to be connected all the time (statistics can also be send to server with each session) the advantage is that the load on the device as network traffic (which is often a constraint) is set to minimum.**

The client has a token of the previous session DEV\_PASS, which is used for the server communication:

This value serves as a proof that the device is the one with which server communicated.

To prevent the direct attack on the KEY\_SET, the KEY\_SET\_KEY is received (for symmetric encryption) from the communication with the server. This value is split into PASS, PIN (permanent), TIME (temporary), DEV\_PASS (calculated from previous values). The value splitting is performed on a server. The task of a client is to receive the TIME and DEV\_PASS from server.

In order to perform the splitting, the server has the knowledge of the user PIN.

The client uses some balanced key exchange to communicate with server:

J-PAKE or SIS-EKE.

The client sends the token to server: DEV\_PASS +PASS+PIN – to prove his identity and identity of the user to server. The server establishes the safe session with the help of new SESSION\_KEY = KEY\_SET\_KEY, verifies the communication and sends the key set to the client.

The online workflow diagram is presented in Figure ZZZZ:

*EKE description*

*Figure ZZZ:* Online workflow diagram.

This proposal uses a light-weighted EKE for server communication like J-PAKE [] or SIS-based EKE []. SIS is a public key encryption that avoids generating the big number of primes and thus can be used for the secure key exchange [], [].

There are generally 3 phases of any EKE:

1. Key establishment (authentication of the user PASS and PIN and device DEV\_PASS)
2. Key verification
3. Sending the data (everytime the TIME should be checked and the data should be send within the protected channel).

In order to clarify the whole process, the J-PAKE protocol [], used for the user credential verification and the key renovation, is briefly described. The protocol acts as soon as the key is expired and the client asks the user to synchronize with the server. Note that the EKE is not demanded to send the files or statistics.

**The client security architecture**

The client architecture consists of the following blocks:

* *Cryptographic functions* that include the decryption (for AES and ABE) key restoration for SSS and EKE token generation procedure.
* *The monitoring infrastructure* that includes simple actors such as the time counter for the key expiry period, the counter of unsuccessful tries in order to protect from the brute force attack, and more elaborate MOS-inspired statistics collector. The alerting and killing functions belong to this block as well.
* *The communication with server* includes the separate sender and receiver to check the user credentials and receiver the new keys which acts based on the EKE. The remaining data like access list and files can be sent via the unprotected channels.
* *The storage* can be separated in two parts: temporary and permanent. While the files and file keys are stored permanently in order to reduce the traffic jam and the resource usage on client, the ABE keys and the key storage protection data are temporary.

The figure XXX illustrates the client security architecture:

**Security analysis**

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

* 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.

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.

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 the bad-willed (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 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 check data does not prevent the user from seeing the contents of decrypted files, but with the wrong password the decrypted files will be totally wrong.

The client still has to count the tries (to prevent the hacker brute force attack) within one session.

In the online mode the protection of the mobile system is backed up by the server. The key expiry period and the usage of J-PAKE guarantees that the sensitive data sent over the network cannot leak. The detailed security analysis of J-PAKE is presented in []. The data sent in the clear does not need to be protected as the AES and ABE guarantee its security.

Additionally, we use the MOS schemes in order to identify anomalous behavior that can indicate an attack and alarm the system of the wrong user behavior. 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 (X)

The selected features shall be modeled as matrices which represents a signal superposition containing noise, legitimate and malicious behavior [18], 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 are 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 steal 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 and correlation analysis of the number of file operations, grouped by the file system location over time, indicate an anomalous behavior;
2. The hacker accesses the mobile client during a valid session and read/write several files at few folders:
   1. The hacker steal 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 and covariance analysis of the number of file operations, grouped by the file system location over time, indicate an anomalous behavior;
3. The hacker emulates a valid session and read/write few files at several folders:
   1. The hacker steal the device;
   2. The hacker changes the device to offline mode;
   3. The hacker changes the device time to a valid period for the current session;
   4. The hacker accesses the mobile client;
   5. The current session is valid;
   6. The hacker perform sparse operations during the valid period;
   7. MOS and correlation analysis of the number of file operations, grouped by the file system location over time, indicate an anomalous behavior;
4. The hacker emulates a valid session and read/write several files at few folders:
   1. The hacker steal the device;
   2. The hacker changes the device to offline mode;
   3. The hacker changes the device time to a valid period for the current session;
   4. The hacker accesses the mobile client;
   5. The current session is valid;
   6. The hacker perform bulk operations during the valid period;
   7. MOS and covariance analysis of the number of file operations, grouped by the file system location over time, indicate an anomalous behavior;

**Complexity analysis**

In the offline mode the client actions are:

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 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 (AES) decryption
6. Modify the TIME periodically ------------------------------ timer
7. Count the tries within the TIME ---------------------------count
8. Modify or delete PUBLIC\_SHARE\_KEY
9. Analyzes the log data and deletes the files when necessary -------------------- TODO

From this analysis, it is possible to observe that the client does not perform complex calculations and does not use too many resources due to the fact that the initial key is shared and the client performs only a decryption, which is not a time-consuming operation. The proposal supports the concept of the light-weighted client, i.e. the most consuming operations are ABE and AES decryption.

In the online mode the client:

1. Generates keys for J-PAKE --- this is a consuming operation. I the future we suggest to replace J-PAKE with SIS-based PKE [].
2. Sends and receives data in the clear.
3. Performs operations 1)-7).
4. Analyses and sends the log data ------- TODO

**Implementation**

The proposed concept of mobile client security has been implemented in the Storgrid protected cloud environment. The mobile client performs the operation both in the offline and online mode and uses the key expiry and ABE to protect itself.

The table below presents the results of processing time testing:

XXX

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

**Conclusion**

**References**

Shamsi, J., Khojaye, M.A., Qasmi, M.A.: Data-intensive

cloud computing: requirements, expectations, challenges,

and solutions. Journal of Grid Computing **11**, 281–310

(2013)

2. 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)

3. Kumar, K., Lu, Y.H.: Cloud computing for mobile users:

Can offloading computation save energy? Computer **43**,

51–56 (2010)

4. Lorido-Botran, T., Miguel-Alonso, J., Lozano, J.A.: A

review of auto-scaling techniques for elastic applications in

cloud environments. Journal of Grid Computing **12**, 559–

592 (2014)

5. 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)

[1] <https://cloudsecurityalliance.org/>

[2] <http://www.ciphercloud.com/blog/cloud-data-security-and-eu-data-privacy-rules-compliance-with-encryption-and-tokenization/>

[3] <http://www.gartner.com/technology/topics/cloud-computing.jsp>

[4] <https://www.skyhighnetworks.com/cloud-university/what-is-cloud-access-security-broker/>

[5] <http://www.gartner.com/technology/reprints.do?id=1-2RUEH70&ct=151110&st=sb>

[6] <http://www.ciphercloud.com/blog/cloud-data-encryption-easy/>

[7] <https://securityintelligence.com/how-to-protect-mobile-apps-essentials/>

[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

[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

[10] TODO: reference to SSS papers

[11] TODO reference to ABE papers

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

[13] 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).

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

[15] 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).

[16] SAP Mobile Platform Secure Mobile with Mocana: webinar whitepaper.

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

[18] 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. Brasilia, 14-16 August 2013. P. 46-51.

[19] 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.[20] Mobile Cyber Threats. Kaspersky Lab & INTERPOL Joint Report. October, 2014. Available: http://media.kaspersky.com/pdf/Kaspersky-Lab-KSN-Report-mobile-cyberthreats-web.pdf

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

[22] 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).