ARDS: A Model for Securing Data in Mobile Cloud Computing

Heba Sherif Sobhy, Amira Mohamed Kotb, Hesham N. Elmahdy

Information Technology Department, Faculty of Computers and Information

Cairo University, Cairo, Egypt.

heba.sherif@fci-cu.edu.eg, a.kotb@fci-cu.edu.eg, ehesham@fci-cu.edu.eg

---

Abstract

For the last few years, mobile usage has grown significantly especially smartphones. Most people decided dispensing the use of their personal computers (i.e. desktops or laptops) and use mobile device instead. Because of mobile limitations, such as limited battery life, limited size of memory, and limited processor's power, the need for Mobile Cloud Computing (MCC) has been increased. The big issue of using MCC is data privacy and security. Data need to be secure while transmission and storage on cloud. This paper proposes a model to secure mobile user's data. This model uses AES, and RSA Digital Signature (ARDS) algorithms. This model provides confidentiality, authentication, and integrity of data stored on mobile cloud. Our model is compared with Garg & Sharma's model that uses RSA, Hashing, and DES algorithm.

Keywords: *Mobile cloud computing, cloud, security, data storage, privacy, confidentiality, integrity, authentication.*

---

1. Introduction

The question asked by anyone dealing with technology andbusiness is, "What is cloud computing, and how it could beuseful for my business?" [1].

Up to now, there is no accepted definition for cloud computing.The most widely used definition is the one defined byThe National Institute of Standards and Technology (NIST),"Cloud computing is a model for enabling ubiquitous, convenient,on-demand network access to a shared pool of configurablecomputing resources (e.g., networks, servers, storage,applications, and services) that can be rapidly provisioned andreleased with minimal management effort or service providerinteraction" [2]. The cloud computing paradigm provides resources to consumers as services. Itis a successful paradigm for deploying web applications[3].The strength of cloudcomputing is because its benefits like scalability, availability,flexibility, minimum cost, location independent, and so on [3,4].The main obstacle for cloud computing that limits its widespread is data privacy and security.

For the last few years, the mobile usage has grownsignificantly especially smartphones. Most people decideddispensing the use of their personal computers (i.e. desktops orlaptops) and use the mobile device instead. While increasingthe use of mobile devices, users want larger storage capacityfor their data. Mobile devices have limited resource suchas limited battery life, limited size of memory, and limitedprocessor's power [5,6]. Because of these limitations, the needfor Mobile Cloud Computing (MCC) has been increased.MCC integrates mobile computing, networking, and cloud computing [7]. By using MCC, most of the processing and datastorage needed by application is moved from the mobiledevice to the

cloud [8]. The big issue of using Mobile CloudComputing is data privacy and security. Data need to besecure while transmission and storage on cloud. Thereis a survey stated that 74% of IT Executives and ChiefInformation Officers are not willing to adopt cloud servicesbecause of the risks associated with security and privacy[8, 9, 10]. That's why several models and mechanismshave been presented by the researchers in the last few years[8].The maintarget of this paper is to provide confidentiality,authentication, and integrity of data stored on the mobilecloud.

The remainder of this paper is organized as follows.Section 2 presents the related work to data security in MCC.The proposed model for data security in MCC is presented insection 3. Section 4 presents the evaluation of the proposedmodel and the results. The conclusion and future works arepresented in section 5.

2. Related Work

In the last few years, many researches and mechanismsabout data security in MCC are presented. Some of them areintroduced in this section.

2.1. A new hybrid encryption protocol (NHEP) for securing data in MCC

Kader et al. [11] proposed a new hybrid encryption protocol(NHEP) to be used for MCC where ECC, AES, RSA,and Blowfish are used for authentication and confidentiality.MD5 is used for integrity. It works as follows:

Encryption Phase: The plaintext is divided into n blocks, each of length128 bits. The first n/2 blocks are encrypted using AES,and ECC encrypts AES secret key. The second n/2blocks are encrypted using Blowfish, and RSA encryptsBlowfish secret key. Hashing using MD5 is applied toboth ciphertexts.

Decryption Phase: The ciphertext is divided into n blocks, each oflength 128 bits. The first n/2 blocks are hashed. Thegenerated hash is compared to the stored one. If thetwo hashes are the same, the first n/2 blocks aredecrypted using ECC and AES. The second n/2 blocksare hashed. The generated hash is compared to thestored one. If the two hashes are the same, the secondn/2 blocks are decrypted using RSA and Blowfish.

2.2. A model for secure sharing of data in MCC usingBlowfish algorithm

Alam et al. [12]proposed a model for secure sharingof data in MCC using Blowfish algorithm. Data ownerencrypts file using Blowfish and sends it to the cloud forstorage. Data sharer downloads the file from cloud, andgets the key of it secretly from data owner to decrypt.Data owner chooses with whom to share his data.

2.3. A model for securing data in MCC using RSA, DES,and Hash Function

Garg et al. [8] proposed a mechanism to provide securityfor data in MCC using RSA algorithm, hash function,and DES algorithm.

2.3.1. Participants
The participants involved in this mechanism are:
- Data Owner (DO): The mobile user.
- Third Party Auditor (TPA): A trusted third party.
- Cloud Service Provider (CSP): The cloud used for storage.

2.3.2. Used Algorithms

RSA: is a widely used public-key cryptosystem(Asymmetric-key Cryptography). It is used for datatransmission, digital signature, key exchange, andencryption/decryption of small data. It uses a keypair (i.e. public, and private keys) of variable size. Tobe secured, key has to be of length more than 1024bits. The practical RSA implementations should usea padding scheme to be secured. The padding schememeans appending some random values to message [13].In RSA with PKCS#1 (v1.5), the most commonlyused padding scheme, data to be encrypted shouldbe less than key length by at least 11 bytes. Forexample, if a key of length 128 bytes (1024 bits) isused, message length should be of length, at most,117 bytes.

Cryptographic Hash Function: produces a message-digestof fixed length for mapping a message ofvariable size. This hash function is unique for eachmessage. The message is padded before computingthe hash function. The most widely used cryptographichash functions are:

- MD5 generates a 128-bit hash value.
- SHA-1 generates a 160-bit hash value.
- SHA-2 generates a hash value of length 224, or256, or 384, or 512 bits [13].

DES: Data Encryption Standard is a symmetric-keycryptography algorithm. It uses a key of length 56bits, and a data block of length 64 bits. Each blockof data is processed in 16 rounds either for encryptionor for decryption [13, 14].

2.3.3. Mechanism

It is divided into four phases as follows:

Key Generation Phase: Both DO and TPA use RSA for generating publicand private keys.

Key Sharing Phase: TPA sends its public key to DO using a secure channel. Then, DO stores it on his/her mobiledevice.

Upload Phase: DO chooses a file F to upload, and encrypts it withhis/her public key using RSA. Then DO generatesa hash function for the encrypted file and encryptsthis hash with TPA's public key using RSA.Finally, DO re-encrypts file with TPA's publickey using RSA, and sends the encrypted file andhash to TPA. TPA stores the encrypted hash anddecrypts the file using its private key. Then TPAgenerates a secret-key using DES, stores it, andencrypts the file using it. Finally, TPA sends theencrypted file to CSP for storage.

Download Phase: DO requests a file from TPA. Then TPA requestsit from CSP. Finally, CSP sends it to TPA. TPAdecrypts the file using the stored DES key, andgenerates a new hash for verifying the integrityof file. Then TPA compares the stored hashwith the generated one, and sends the resultwhich indicates file's correctness to DO with thefile. DO checks the file's correctness, and thendecrypts it with his/her private key using RSA.

3. Proposed Model (ARDS)

Our proposed model uses AES, and RSA Digital Signaturealgorithms to provide data security in MCC. The proposedarchitecture of ARDS model is shown in Figure 1.
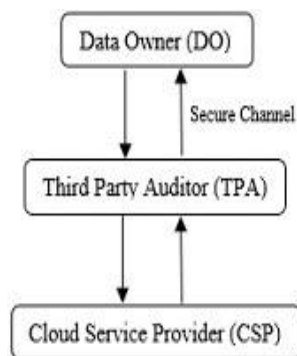
Figure 1. ARDS architecture

3.1. Participants

The participants involved in this mechanism are:

- Data Owner (DO): Mobile user who wants to storehis/her data on mobile cloud.
- Third Party Auditor (TPA): A trusted third party thatdo some processing instead of mobile.
- Cloud Service Provider (CSP): Provides storage servicesto mobile users.

3.2. Used Algorithms

This model uses RSA digital signature for authenticationand integrity of files. It uses AES to provide confidentiality by encrypting/decrypting files.

RSA Digital Signature:RSA algorithm is discussed in section 2.3.2. RSA DigitalSignature is the most widely used digital signaturetechnique. It is used for authentication and messageintegrity. The algorithm generally works as shown inFigure 2 [13]. RSA digital signature generally depends on RSA cryptography and message hashing[15]. The message has to be firstly hashed, then this hash is signed by the sender using RSA[13, 15].
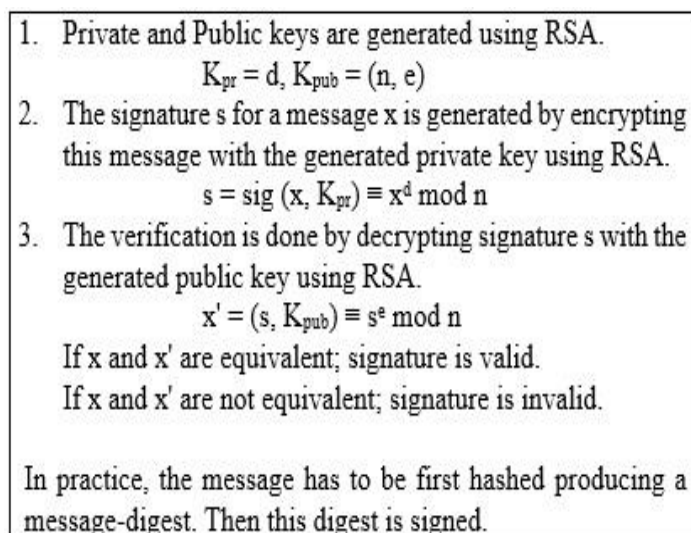

1. Private and Public keys are generated using RSA.
$$K_{pr} = d, K_{pub} = (n, e)$$
2. The signature s for a message x is generated by encrypting this message with the generated private key using RSA.
$$s = sig (x, K_{pr}) \equiv x^d \bmod n$$
3. The verification is done by decrypting signature s with the generated public key using RSA.
$$x' = (s, K_{pub}) \equiv s^e \bmod n$$
If x and x' are equivalent; signature is valid.
If x and x' are not equivalent; signature is invalid.

In practice, the message has to be first hashed producing a message-digest. Then this digest is signed.

Figure 2. RSA digital signature

AES: Advanced Encryption Standard (AES), also known asRijndael, is a symmetric-key block cipher. AES isadopted by the U.S. government, and now used by thewhole world. It

uses a 128-bit block size, with keyof sizes 128, 192, or 256 bits. Each block of data isprocessed in a number of rounds either for encryptionor for decryption. All the rounds do the same operation.The number of rounds differs according to the key size as shown in Table 1. In 1997, the National Instituteof Standards and Technology (NIST) announced that itneeds a successor for DES and Triple-DES algorithms.This successor can provide data security for 20-30years and to be used worldwide. AES proves itself andbecame the most popular symmetric-key cryptographicalgorithm over DES and 3DES. AES is more secure,and faster than both DES and 3DES [13, 14].

Table 1. No. of rounds and key sizes for AES

Key Size	Number of rounds
128-bit	10
192-bit	12
256-bit	14

3.3. Mechanism

This mechanism is divided into four phases as follows:

Key Generation Phase: DO uses RSA for generating his/her public and privatekeys to be used for digital signature. Then he/shegenerates a 128-bit secret key using AES for encryptingfiles.

Key Sharing Phase: DO sends his/her public key to TPA using a securechannel.

Upload Phase: DO chooses a file F to upload, and encrypts it withhis/her secret key using AES. Then DO generates a hashfunction for the encrypted file, and signs this hash withhis/her private key using RSA digital signature. Finally,DO sends the encrypted file and signature to TPA.TPA checks authentication and integrity of file by generatinga new hash, and verifying the signature with DO'spublic key. Then TPA compares the verified signature(i.e. old hash) with the generated one, and stores thesignature. Then TPA generates a new 128-bit secret-keyfor file F using AES, stores it, and encrypts the fileusing it. Finally TPA sends the encrypted file to CSPfor storage.

Download Phase: DO requests a file from TPA. Then TPA requests it from CSP. Finally, CSP sends it to TPA. TPA decrypts file using the stored secret key. Then TPAchecks authentication and integrity of file by generatinga new hash, verifying the signature with DO's publickey, and compares the verified signature (i.e. old hash)with the generated one. Finally, TPA sends the result ofcomparison which indicates the correctness of file withthe file to DO.DO checks the correctness of file, and then decrypts itwith his/her secret key using AES. The mechanism is summarized in Figure 3.
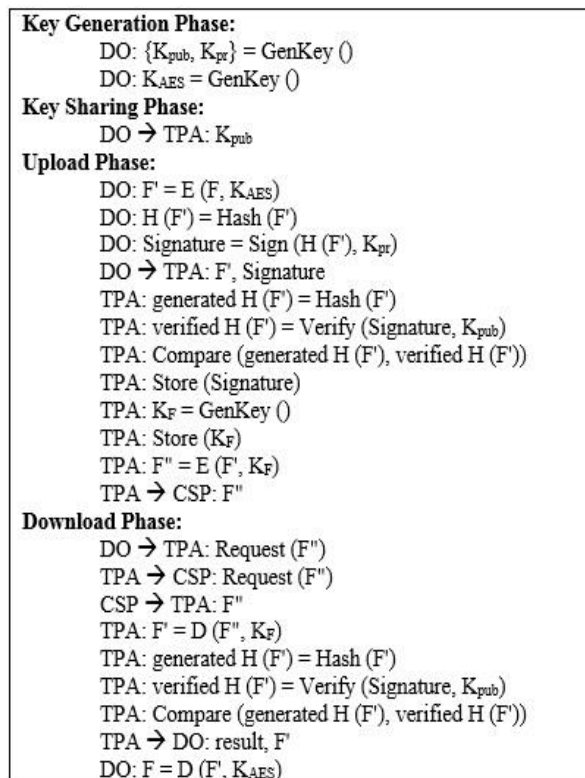
```
Key Generation Phase:
       DO: {Kpub, Kpr} = GenKey ()
       DO: KAES = GenKey ()
Key Sharing Phase:
       DO → TPA: Kpub
Upload Phase:
       DO: F' = E (F, KAES)
       DO: H (F') = Hash (F')
       DO: Signature = Sign (H (F'), Kpr)
       DO → TPA: F', Signature
       TPA: generated H (F') = Hash (F')
       TPA: verified H (F') = Verify (Signature, Kpub)
       TPA: Compare (generated H (F'), verified H (F'))
       TPA: Store (Signature)
       TPA: KF = GenKey ()
       TPA: Store (KF)
       TPA: F" = E (F', KF)
       TPA → CSP: F"
Download Phase:
       DO → TPA: Request (F")
       TPA → CSP: Request (F")
       CSP → TPA: F"
       TPA: F' = D (F", KF)
       TPA: generated H (F') = Hash (F')
       TPA: verified H (F') = Verify (Signature, Kpub)
       TPA: Compare (generated H (F'), verified H (F'))
       TPA → DO: result, F'
       DO: F = D (F', KAES)
```

Figure 3. ARDS mechanism

4. Evaluation and Results

In the previous section a model for securing data in MCCwas introduced. The main target of it is to provide confidentiality, authentication, and integrity for data stored on the mobile cloud. The model, named ARDS, uses AES andRSA digital signature. It is compared with Garg & Sharma'smodel which was introduced in section 2.3. This section states the environment used for the development, securityanalysis, computational overhead, storage requirements,processing time, and data overhead of both models.

4.1. Environment

The model consists of data owner, third party auditor, andcloud service provider. Data owner is a mobile device withAndroid 5.0.2 (Lollipop) operating system, Quad-core1.5 GHz processor, and with 3 GB RAM. Third partyauditor uses Windows 7, 64-bit operating system Intel(R)Core(TM) i5, 2.4 GHz processor, and with 4 GB RAM.TPA is simulated in JAVA. For the cloud, dropbox is used.Padded RSA with PKCS#1 (v1.5) is used with 1024-bit(128-byte) keys. AES algorithm is used with 128-bit (16-byte) key. The cryptographic hashing algorithm usedis SHA-2 with length 256 bits (32 bytes).

4.2. Security Analysis

4.2.1. Garg & Sharma's Model

This model provides confidentiality of datastored on the mobile cloud. But it does not provideauthentication and integrity of data in a correct way.

Confidentiality: Confidentialityof data is provided during data transmissionand data storage. The file is encrypted while transmissionfrom DO to TPA, and from TPA to CSP. Thisavoids any intruder from knowing any informationfrom the transferred file. The file

stored on the cloudis encrypted twice; once with DO's key, and oncewith a key generated specifically for this file by TPA.This avoids CSP and anyone access file on the cloudfrom knowing the content of this file.

Authentication and Integrity: The file and the hashof file are signed by TPA's public key. Any intruder(e.g. other user) knows TPA's public key can changefile, computes a hash for the new file, and thensigns the new file and hash using TPA's public key.Moreover, TPA does not verify integrity of file beforesending it to the cloud for storage. Integrity of file isverified only while downloading file from the cloud.

4.2.2. ARDS Model

This model provides confidentiality, authenticationand integrity of data stored on the mobile cloud.

Confidentiality: Confidentiality of data is provided while transmission andstorage of data. The file is encrypted when it istransmitted from DO to TPA, and from TPA toCSP. This avoids any intruder from knowing anyinformation from the transferred file. The file storedon the cloud is encrypted with DO's key, and then re-encryptedwith the key generated by TPA specificallyfor this file. This avoids CSP and anyone access fileon the cloud from knowing the content of this file.

Authentication and Integrity: The file is hashedand this hash is signed by DO's private key. Nointruder can change file while transmission becauseno one know the private key of DO to generate ahash and sign it. TPA verifies authentication andintegrity of file before sending it to the cloud forstorage, and again while downloading it from thecloud.

4.3. Computational Overhead

4.3.1. Garg & Sharma's Model

The computational overhead of Garg & Sharma's modelaccording to phases is as follows:

Key Generation Phase: Both DO, and TPA performspublic and private keys generation.

Upload Phase: DO performs two encryption for file,one hash function generation, and one encryption forthe hash. While TPA performs one decryption forfile, one key generation, and one encryption for file.

Download Phase: DO performs one decryption forfile. While TPA performs one decryption for file, onehash function generation, and one decryption for thehash.

The computational overhead of this model is summarized in Table 2.

Table 2. Computational overhead of Garg & Sharma's model

	Cryptography	Hashing	Key Generation
Key Generation Phase			
DO	0	0	1
TPA	0	0	1
Upload Phase			
DO	3	1	0
TPA	2	0	1
Download Phase			
DO	1	0	0
TPA	2	1	0

4.3.2. ARDS Model

The computational overhead of ARDS modelaccording to phases is as follows:

Key Generation Phase:DO performs public andprivate keys generation. Moreover, he/she generatesa secret key for files.

Upload Phase:DO performs one encryption for file,one hash function generation, and one encryption forthe hash. While TPA performs one hash functiongeneration, one decryption for the hash, one keygeneration, and one encryption for file.

Download Phase: DO performs one decryption forfile. While TPA performs one decryption for file, onehash function generation, and one decryption for thehash.

The computational overhead of this model is summarizedin Table 3.

Table 3. Computational overhead of ARDS model

	Cryptography	Hashing	Key Generation
Key Generation Phase			
DO	0	0	2
TPA	0	0	0
Upload Phase			
DO	2	1	0
TPA	2	1	1
Download Phase			
DO	1	0	0
TPA	2	1	0

4.4. Storage Requirements
4.4.1. Garg & Sharma's Model

The storage required by Garg & Sharma's modelaccording to participants is as follows:

- DO stores his/her public and private keys, and the public key of the TPA.
- TPA stores its public and private keys, an encrypted hash for each file, and a secret key for each file.
- CSP stores encrypted files uploaded by mobile user.

The storage requirements of this model is summarized in Table 4.

Table 4. Storage requirements of Garg & Sharma's model

Participants	Storage Requirements
DO	1. His/her public and private keys. 2. TPA's public key.
TPA	1. Its public and private keys. 2. **For each file:** encrypted hash of file. 3. **For each file:** secret key.
CSP	Encrypted files uploaded by DO.

4.4.2. ARDS Model

The storage required byARDS modelaccording to participants is as follows:

- DO stores his/her public key, private key, and secretkey for files.
- TPA stores public key of DO, a signature (encryptedhash) for each file, and a secret key for each file.
- CSP stores encrypted files uploaded by mobile user.

The storage requirements of this model is summarized in Table 5.

Table 5.Storage requirements of ARDS model

Participants	Storage Requirements
DO	1. His/her public and private keys. 2. His/her secret key for files.
TPA	1. DO's public key. 2. **For each file:**signature of file. 3. **For each file:** secret key.
CSP	Encrypted files uploaded by DO.

4.5. Processing Time

Both Garg & Sharma's model, and ARDS model areimplemented in the same environment stated above. Theprocessing time is measured in seconds. Each time valueis the average of 5 runs done in different times. The averageprocessing time is calculated for each phase as follows:

Key Generation Phase: The average processing timeneeded by each model in this phase is shown in Table 6 and Figure 4. It is clear that Garg & Sharma'smodel is slower than ARDS model. DO's processingin Garg & Sharma's model is slightly faster than thatof ARDS model because DO generates an asymmetric-keypair only, but in ARDS model he/she generates anextra symmetric-key. In ARDS model there is no TPA'sprocessing.

Table 6.Average processing time in key generation phase

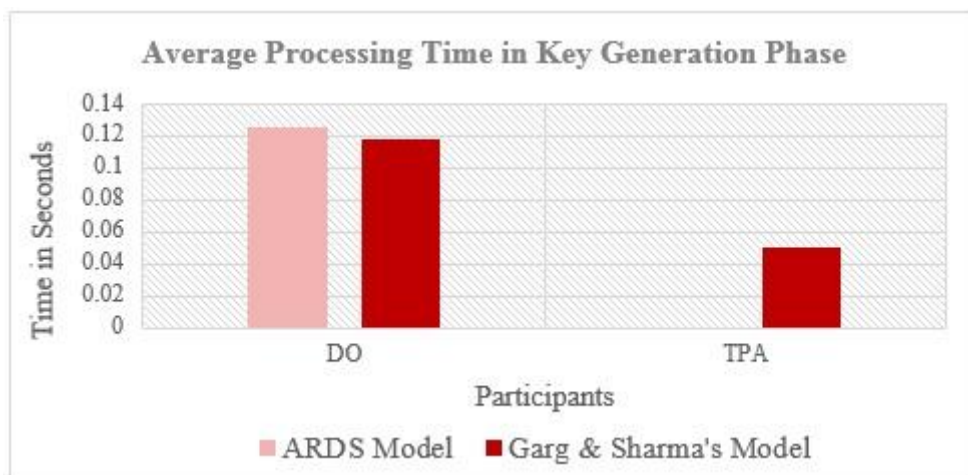| Garg & Sharma's Model			ARDS Model		
DO	**TPA**	**Total**	**DO**	**TPA**	**Total**
0.118	0.05	0.168	0.125	0	0.125

Figure 4. Average processing time in key generation phase

Upload Phase: DO's average processing time needed byeach model in upload phase is shown in Table 7 and Figure 5. It is clear from them that Garg & Sharma's modelis the slower. The main reason is that encrypting largedata with asymmetric algorithm like RSA is not preferredyet as it takes too long time. Moreover, DO encrypts fileusing RSA twice. That's why there is a great differencebetween the processing of the two models especiallywhen file is large (i.e. files of sizes 1 MB, 2 MB, and5 MB). TPA's average processing time needed by eachmodel in upload phase is shown in Table 8 and Figure 6.It is clear that Garg & Sharma's model is the slower. Themain reason is that TPA decrypts file using RSA beforeencrypting it with DES and this takes too long time.

It is clear from all of the above that ARDS model is fasterthan Garg & Sharma's model in uploading files. Thewhole average processing time needed by both modelsto upload files is summed up and summarized in Table 9.

Table 7. DO's average processing time in upload phase

Model File Size	Garg & Sharma's Model	ARDS Model
117 bytes	0.051	0.024
128 bytes	0.053	0.025
512 bytes	0.057	0.026
1 KB	0.059	0.027
5 KB	0.155	0.029
10 KB	0.296	0.032
50 KB	0.759	0.042
100 KB	2.666	0.059
1 MB	26.251	0.357
2 MB	52.588	0.626
5 MB	132.18	1.496

Table 8. TPA's average processing time in upload phase

Model / File Size	Garg & Sharma's Model	ARDS Model
117 bytes	0.071	0.036
128 bytes	0.074	0.039
512 bytes	0.074	0.046
1 KB	0.087	0.048
5 KB	0.185	0.052
10 KB	0.209	0.055
50 KB	0.733	0.059
100 KB	1.372	0.075
1 MB	13.159	0.107
2 MB	26.719	0.163
5 MB	64.594	0.279
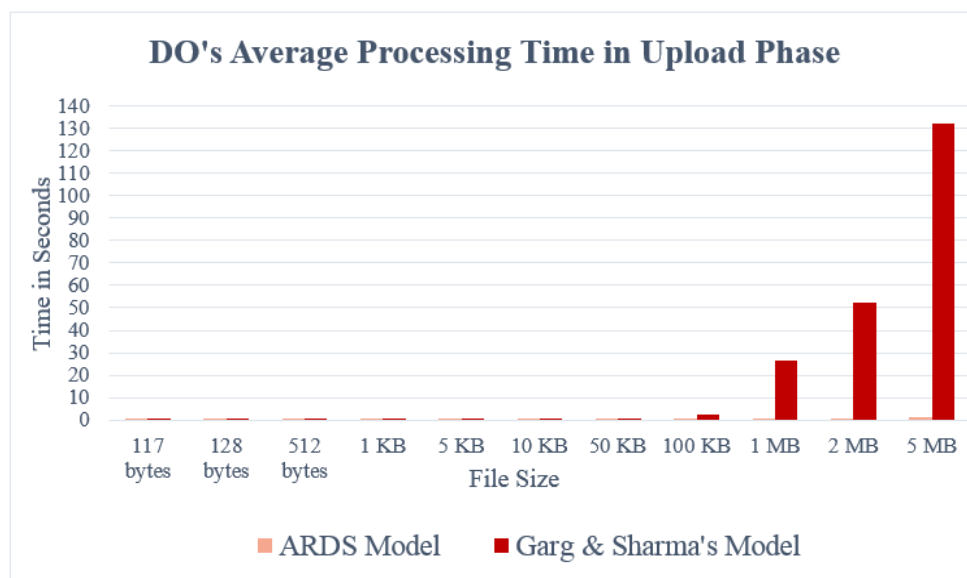
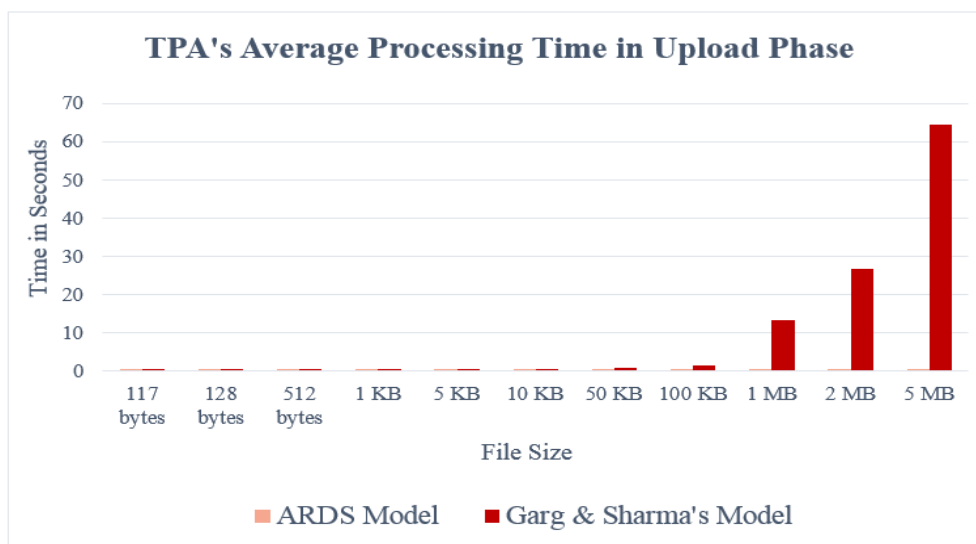**Figure 5. DO's average processing time in upload phase**

Figure 6. TPA's average processing time in upload phase

Table 9. Average processing time in upload phase

Model File Size	Garg & Sharma's Model			ARDS Model		
	DO	TPA	Total	DO	TPA	Total
117 bytes	0.051	0.071	0.122	0.024	0.036	0.06
128 bytes	0.053	0.074	0.127	0.025	0.039	0.064
512 bytes	0.057	0.074	0.131	0.026	0.046	0.072
1 KB	0.059	0.087	0.146	0.027	0.048	0.075
5 KB	0.155	0.185	0.34	0.029	0.052	0.081
10 KB	0.296	0.209	0.505	0.032	0.055	0.087
50 KB	0.759	0.733	1.492	0.042	0.059	0.101
100 KB	2.666	1.372	4.038	0.059	0.075	0.134
1 MB	26.251	13.159	39.41	0.357	0.107	0.464
2 MB	52.588	26.719	79.307	0.626	0.163	0.789
5 MB	132.18	64.594	196.774	1.496	0.279	1.775

Download Phase: DO's average processing time neededby each model in download phase is shown in Table 10 and Figure 7. It is clear from them that Garg & Sharma'smodel is the slower because of decrypting large datawith RSA. TPA's average processing time needed by eachmodel in download phase is shown in Table 11 and Figure 8. It is clear from them that Garg & Sharma's modelis slightly slower because of decrypting the stored hashusing RSA.

It is clear from all of the above that ARDS modelis the faster in downloading files. The whole averageprocessing time needed by both models to downloadfiles is summed up and summarized in Table 12.

Table 10. DO's average processing time in download phase

Model File Size	Garg & Sharma's Model	ARDS Model
117 bytes	0.026	0.014
128 bytes	0.03	0.014

512 bytes	0.036	0.017
1 KB	0.049	0.018
5 KB	0.098	0.022
10 KB	0.115	0.026
50 KB	0.343	0.032
100 KB	0.63	0.044
1 MB	5.874	0.301
2 MB	11.883	0.487
5 MB	29.55	1.289

Table 11. TPA's average processing time in download phase

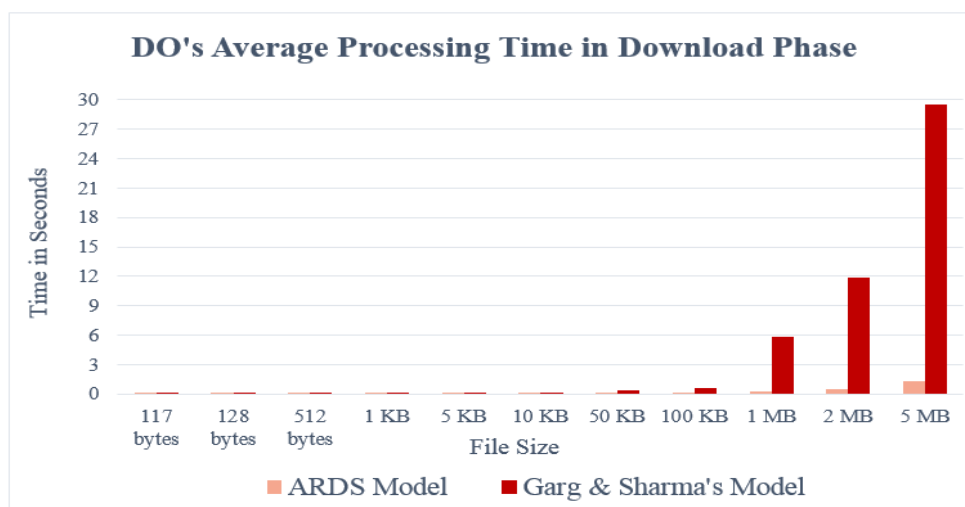| Model \ File Size	Garg & Sharma's Model	ARDS Model
117 bytes	0.072	0.029
128 bytes	0.09	0.031
512 bytes	0.111	0.037
1 KB	0.125	0.041
5 KB	0.14	0.05
10 KB	0.19	0.055
50 KB	0.23	0.059
100 KB	0.257	0.064
1 MB	0.295	0.118
2 MB	0.329	0.135
5 MB	0.397	0.212

Figure 7. DO's average processing time in download phase
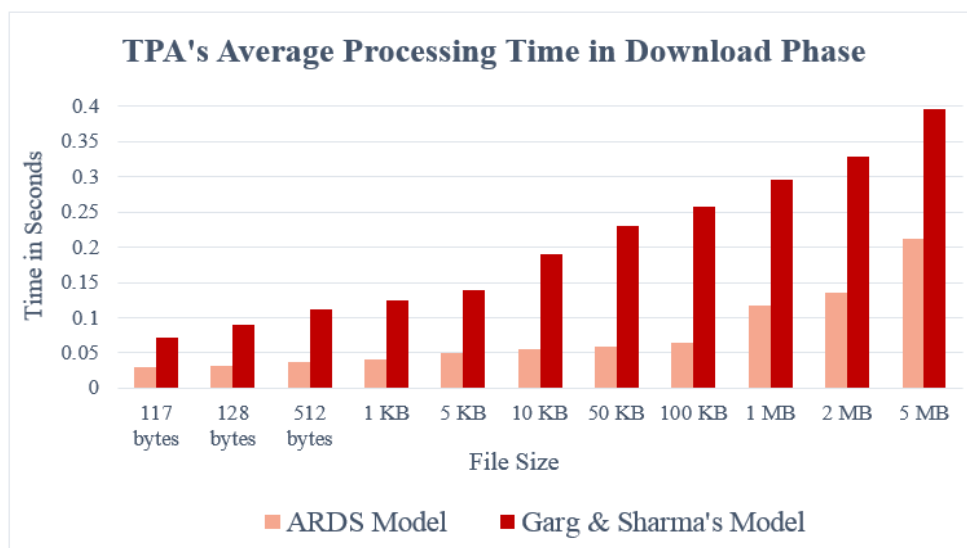
Figure 8. TPA's average processing time in download phase

Table 12. Average processing time in download phase

Model File Size	Garg & Sharma's Model			ARDS Model		
	DO	TPA	Total	DO	TPA	Total
117 bytes	0.026	0.072	0.098	0.014	0.029	0.043
128 bytes	0.03	0.09	0.12	0.014	0.031	0.045
512 bytes	0.036	0.111	0.147	0.017	0.037	0.054
1 KB	0.049	0.125	0.174	0.018	0.041	0.059
5 KB	0.098	0.14	0.238	0.022	0.05	0.072
10 KB	0.115	0.19	0.305	0.026	0.055	0.081
50 KB	0.343	0.23	0.573	0.032	0.059	0.091
100 KB	0.63	0.257	0.887	0.044	0.064	0.108
1 MB	5.874	0.295	6.169	0.301	0.118	0.419
2 MB	11.883	0.329	12.212	0.487	0.135	0.622
5 MB	29.55	0.397	29.947	1.289	0.212	1.501

4.6. Data Overhead

The size of encrypted files stored on the cloud is comparedin both models. This comparison is shown in Table 13, and the data overhead (i.e. added bytes) is shown in Figure 9. It isclear from them that file size is enlarged inGarg & Sharma's model. The main reason is that this modeluses padded RSA to be secured. Padded RSA with 128-byte key length chunks data into 117-byte block length.Each block is padded to be of length 128 bytes. While inARDS model; only the last block of data will be paddedwhen file is not dividable by block size.

Table 13. Size of encrypted files in both models

Model File Size	Garg & Sharma's Model	ARDS Model
117 bytes	128 bytes	128 bytes
128 bytes	256 bytes	128 bytes
512 bytes	640 bytes	512 bytes
1 KB	1.125 KB	1 KB
5 KB	5.5 KB	5 KB
10 KB	11 KB	10 KB
50 KB	54.75 KB	50 KB
100 KB	109.5 KB	100 KB
1 MB	1.094 MB	1 MB
2 MB	2.188 MB	2 MB
5 MB	5.47 MB	5 MB
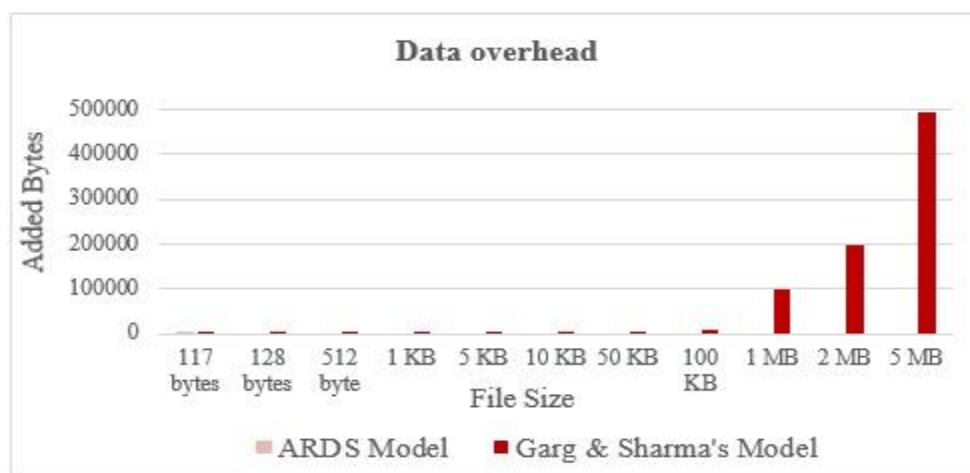
Figure 9. Data overhead in both models

5. Conclusion and Future Works

The main objective of this paperis to propose a model for securing data in mobile cloud computing.This model should take into account the mobile device's limitations such as, battery life, processor's power, and memory size.A number of frameworks were implemented to secure data in mobile cloud computing. Some of these frameworks were presented here, and one of them was chosen to evaluate our model according to it. The chosen model was introduced by Garg and Sharma in[8]. Garg & Sharma's model uses Hashing, RSA, and DES algorithms to secure data in MCC.Our proposed model, named ARDS, uses AES and RSA digital signature to provide authentication, confidentiality, and integrity of data stored on the mobile cloud. By using this model, users can save their data regardless its size, and without caring about privacy and security.

A simulation program was developed for both models (Garg & Sharma's model and the proposed ARDS model). The two models were compared, and the result was that ARDS model provides better performance, storage, and security than Garg & Sharma's model because of the following:

1. Garg & Sharma's model signs the file and hash of file using TPA's public key which can be known easily. It is known to other users. While ARDS model signs the hash of file using DO's private key which is known only to that DO.

2. Garg & Sharma's model doesn't verify integrity of file before sending it to the cloud for storage. It is verified only after downloading file. While ARDS model verifies integrity of file before sending it to the cloud and after downloading it from the cloud.

3. Garg & Sharma's model uses RSA algorithm for the encryption and decryption of files. Encrypting or decrypting large data with asymmetric algorithm like RSA is not preferred yet as it takes too long time. The file is encrypted twice on behalf of the mobile device using RSA. While ARDS model encrypts and decrypts files using AES algorithm which is fast and much secured symmetric algorithm. The file is encrypted only one time on behalf of the mobile device.

4. Garg & Sharma's model uses RSA for encrypting and decryption of files and using such algorithm should be accompanied with a padding scheme which means enlargement of ciphertext. For example, in RSA with PKCS#1 (v1.5), the most commonly used padding scheme, data to be encrypted should be less than key length by at least 11 bytes (i.e. If we use a key of size 128 bytes, data is chunked into 117-byte block length. Each block is padded to be of length 128 bytes). While ARDS model uses AES which results in a ciphertext nearly equal to the plaintext except if plaintext is not dividable by block size, the last block will be padded.

Data security in mobile cloud computing is a huge research area. There are several challenges and issues should be addressed to help the deployment of mobile cloud computing. Some of the future works are proposed as follows:

1. Reducing the processing done on behalf of the mobile device more by trying well-secured algorithms other than AES, and RSA digital signature.

2. The mobile device can be lost easily, so we need to find a way to secure storage of keys (i.e. Recovery and backups of keys).

3. The mobile can be stolen easily at any time so, we need to add some additional authentication for the data owner. We can use passwords, one time passwords (OTP), biometrics, and so on.

References

[1]. S. Zhang, X. Chen, S. Zhang,and X. Huo, "The comparison between cloud computing and grid computing,"In Proc. Int. Conf. on Computer Application and System Modeling (ICCASM), IEEE,vol.11,2010,pp.72-75.

[2]. P. Mell and T. Grance, "The nist definition of cloud computing," 2011. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/ 800-145/SP800-145.pdf

[3]. N. A. Abd-El Azim, and A. H. El Bastawissy, "Transactions management in cloud computing," Egyptian Computer Science Journal, vol.38, no.1, 2014.

[4]. A. Ahmad, M. M. Hassan, and A. Aziz, "A multi-token authorization strategy for secure mobile cloud computing," In Proc. Int. Conf. on Mobile Cloud Computing, Services, and Engineering(MobileCloud), IEEE, 2014, pp.136-141.

[5]. D. Jana, and D.Bandyopadhyay, "Management of security and privacy issues of application development in mobile cloud environment: A survey," In Proc. Int. Conf. on Recent Advances and Innovations in Engineering(ICRAIE), IEEE, 2014, pp.1-6.

[6]. S. Abolfazli, Z. Sanaei, M. H. Sanaei, M. Shojafar, and A. Gani, "Mobile cloud computing: The-state-of-the-art, challenges, and future research," 2015.

[7]. N. Aminzadeh, Z. Sanaei, and S. H. A. Hamid, "Mobile storage augmentation in mobile cloud computing: Taxonomy, approaches, and open issues," Simulation Modelling Practice and Theory, vol.50, 2015, pp. 96-108.

[8]. P. Garg, and V. Sharma, "An efficient and secure data storage in mobile cloud computing through RSA and Hash function," In Proc. Int. Conf. on Issues and Challenges in Intelligent Computing Techniques (ICICT), IEEE, 2014, pp.334-339.

[9]. S. Subashini, and V. Kavitha. "A survey on security issues in service delivery models of cloud computing," Journal of Network and Computer Applications, vol.34, no.1, 2011, pp.1-11.

[10]. A. N. Khan, M. L. M. Kiah, S. U. Khan, and S. A. Madani, "Towards secure mobile cloud computing: A survey." Future Generation Computer Systems, vol.29, no.5, 2013, pp. 1278-1299.

[11]. H. M. A. Kader,M. M. Hadhoud, S. M. El-Sayed, and D. S. AbdElminaam, "Performance evaluation of new hybrid encryption algorithms to be used for mobile cloud computing," International Journal of Technology Enhancements and Emerging Engineering Research, vol.2, no.4, 2014.

[12]. M. M. Alam, S. Hati, D. De, and S. Chattopadhyay, "Secure sharing of mobile device data using public cloud," In Proc. of CONFLUENCE,Fifth International Conference on Confluence The Next Generation Information Technology Summit, IEEE, 2014, pp.149-154.

[13]. C. Paar, and J. Pelzl,"Understanding cryptography: a textbook for students and practitioners,"Berlin, Germany, Springer verlage, 2010.

[14]. W. Stallings,"Cryptography and network security: principles and practices," Edition 4,Prentice Hall, 2006.

[15]. K. H. Rahouma, "Design of a multi-digital signature protocol," Egyptian Computer Science Journal, vol.28, no.1, 2006.