International Research Journal of Computer Science (IRJCS)
Issue 01, Volume 5 (January 2018)
ISSN: 2393-9842
www.irjcs.com

ASYMMETRIC AND SYMMETRIC CRYPTOGRAPHY TO SECURE SOCIAL NETWORK MEDIA COMMUNICATION : THE CASE OF ANDROID-BASED E-LEARNING SOFTWARE

Anis Cherid
Computer Science Faculty, Universitas Mercu Buana,
Jakarta 11650, Indonesia
anis.cherid@mercubuana.ac.id

Manuscript History
Number: **IRJCS/RS/Vol.05/Issue01/JACS10080**
DOI: 10.26562/IRJCS.2018.JACS10080
Received: 02, December 2017
Final Correction: 22, December 2017
Final Accepted: 04, January 2018
Published: January 2018
Citation: Anis, C. (2018). ASYMMETRIC AND SYMMETRIC CRYPTOGRAPHY TO SECURE SOCIAL NETWORK MEDIA COMMUNICATION: THE CASE OF ANDROID-BASED E-LEARNING SOFTWARE. IRJCS:: International Research Journal of Computer Science, Volume IV, 01-08. **doi: 10.26562/IRJCS.2017.JACS10080**
Editor: Dr.A.Arul L.S, Chief Editor, IRJCS, AM Publications, India

Abstract—"Pemahan Omah @UMB" is an Android-based server-less e-learning software that uses social network media as its main communication channel to avoid the burden of maintaining a course management system server and its supporting infrastructure. In the first version of this software, there are still some limitations. The first one is that it is securing exchanged data only in the form of "security by obscurity" and the second one is that it can only send plain text message without any images attached in the message. To eliminate these limitations, several new features are employed in the new version of the software. These features are creating message as file attachment that can contain text and messages, securing a large file with the combination of symmetric AES cryptographic algorithm and asymmetric RSA cryptographic algorithm, and building public-private key generation and management. The RSA cryptography and AES cryptography used in the new version of the e-learning software is implemented by using the "javax.crypto" library, which is readily available in the Android's Java development environment.

Keywords—Information security; server-less e-learning software; course management software; social network application; RSA cryptography; AES cryptography;

I. INTRODUCTION

A. Background and Motivation

One of the communication channels that have been overlooked for inter-applications communication is the social network media. Social network media is an open communication platform, so the sent message must be protected from any kind of modification. This issue is even more important when the application must preserve the originality of the message. In the case of e-learning software for example, after a student has finished working on a quiz, the application must send the result or the grade of the student to the teacher. If the quiz result is sent through a social network application, then it must be protected from any kind of modification. Most social network applications do not provide any API (Application Programming Interface), so it is usually not easy to make two applications to communicate automatically through this channel.

The applications need the help of a human being to communicate. It is more obvious now, that the message produced by the applications, must be kept in the form that is easy to copy-and-paste or easy to forward to the social network media by a person. “Pemahan Omah @UMB” is an Android-based server-less e-learning system, developed by the author for Computer Science Faculty of Universitas Mercu Buana Indonesia and for Training and Development Center for Pre-School Education and Community Education of DKI Jakarta Province. It can support the process of distance learning without using any Course Management System server (CMS server) and its infrastructure, because the software leverages social media network as its main communication channel. The first version of the software still has some limitations. One of them is that the software uses symmetric cryptographic algorithm to protect the process of sending message between parties involved in the learning process, but only implements a rudimentary "security by obscurity" symmetric key management. The second limitation is that the software can only send limited size of plain text messages and it is not possible to embed pictures in the messages.

B. Research Questions

In this paper the *following* research questions will be answered:

1. How to implement asymmetric cryptographic algorithm by using the Android device or smartphone, so that every message exchanged between the parties involved can be prevented from falling into the wrong hands and can be secured from unintended alteration.
2. How to analyze, design, and implement asymmetric cryptographic algorithm, so two applications can communicate between each other in the Android platform by using the social network application installed in the device.
3. How to analyze, design, and implement asymmetric cryptographic algorithm in the case of Android-based server-less e-learning application, to secure the exchange of messages between teachers and students trough the social network application.

C. Research Methods

Based on the research questions described before, the methods used in this paper are the followings:

1. Identify the main questions that must be answered in in this research. This research is trying to answer the question on how to develop messaging system between a teacher and the students of an e-learning environment, by using asymmetric cryptographic algorithm. The message is produced by the e-learning software running on the teacher's Android device and will be sent through the social network application that runs on the teacher's device, to the students' device. The process of sending the message through the social network application is done manually by the teacher. Later, the message will be imported to the e-learning software running on the students' Android device. The process of getting the message from the social network application to the e-learning system is done manually by the students.
2. Search for the relevant literature related to the implementation of asymmetric cryptographic algorithm in the Android platform.
3. Analyze, design, implement, and test the software that enables the complete business process and algorithm to exchange message securely between applications, through the social network application in the Android platform.

II. LITERATURE REVIEW

A. Cryptographic Algorithms

There are three types of cryptographic algorithms: secret key algorithms (symmetric algorithms), public key algorithms (asymmetric algorithms), and hashing algorithms (one-way algorithms). In secret key algorithms, both participants in the communication share a single key. One example of secret key algorithms is the AES (Advanced Encryption Standard) [1].

Asymmetric cryptography or public key cryptography is a cryptography that is not using the same secret key to encrypt and decrypt a message. It is using a pair of keys, a public key and a private key, owned by each participant in the process of exchanging messages. The public key is distributed and known by everybody, while the private key is kept secret by its owner. An example of public key algorithms is the RSA (Rivest Shamir Adleman). RSA is grounded in number theory and its security comes from the premise that factoring large numbers is a computationally expensive proposition [1].

Because of its computation complexity, RSA algorithm is usually used to secure a small size block of data, mostly to secure and send symmetric cryptography key, e.g. the session key. The symmetric key is then used to apply cryptography to the subsequent longer blocks of data. Symmetric encryption/decryption can be done with simpler algorithm and is done much faster than its asymmetric counterpart and because of that, the whole process of securing data by using the combination of asymmetric and symmetric algorithm is a better choice than using the asymmetric algorithm alone [1]. Public key cryptograhy will achieve both encryption and authentication, if message encryption is done by using both the public key of the receiver and the private key of the sender.

Encryption is achieved because message is encrypted using the public key of the receiver and can only be decrypted by using the receiver private key, while authentication is achieved because message is also encrypted by using the private key of the sender and the identity of the sender is proved because the message can only be decrypted by using the public key of the sender [2][3].

The third cryptographic algorithm is a hash or message digest function. This algorithm is meant to compute a cryptographic cheksum over a message and there is no computationally feasible way to get the original message. The checksum can be used to save the hash value of a password into the permanent storage, so a password can be validated by using the checksum and without compromising the original password. The checksum can also be used to create a digital signature of a message, so the message receiver can check if the message has been or has not been tempered [1][3]. An example of this algorithm is the SHA (Secure Hash Algorithm) [2].

B. Cryptography Library in Android's Java Development Environment

The Java Development Kit (JDK) has included programming library to implement symmetric, asymmetric, and hashing cryptographic algorithm. When the JDK is adapted to the Android system, most of the functions in the library are also included. In the Android system, the library to implement the cryptography system is available from the "javax.crypto" package [4].

The steps to implement asymmetric algorithm that provide both encryption/decryption feature and authentication feature are [2]:

1. A person who wants to receive a message (*person R*) generate a pair of public key (*public key R*) and private key (*private key R*). Then, *person R* sends the generated *public key R* to a person who will send a message (*person S*).
2. The person who will send a message (*person S*) generate a pair of public key (*public key S*) and private key (*private key S*). Then, *person S* sends *public key S* to the person who wants to receive a message (*person R*).
3. The message sender, who is *person S,* encrypts a message with the public key of *person R,* that is *public key R.* To authenticate the sender's identity, *person S* further encrypts the message with the private key of himself/herself, that is *private key S.* The message that has been encrypted twice with *public key R* and then with *private key S*, is then sent to *person R.*

To get the original message sent by person S, person R decrypts the message with public key S and then decrypts it the second time with his/her own private key, which is private key R [2].

To accomplish the above-mentioned steps with the "RSA" cryptographic algorithm, Android system's cryptography library ("javax.crypto" package) provides several simple-to-use classes and methods [4]:

1. To generate public key-private key pair, the classes are **KeyPairGenerator** and **KeyPair**, while the methods are **KeyPairGenerator.getInstance()**, **KeyPairGenerator.getInstance().initialize()**, and **KeyPair-Generator.getInstance().generateKeyPair().**
2. To get and then convert the public key and private key into byte format that can be easily saved to permanent storage, the class is **KeyPair** and the methods are **getPublic().getEncoded()** and **getPrivate().getEncoded()**.
3. To convert byte format of public key and private key read from the permanent storage, back into key specifications (and then later back into public key and private key), the class is **X509EncodedKeySpec**.
4. To convert key specifications back into public key and private key, the classes are **KeyFactory**, **PublicKey**, and **PrivateKey**, while the methods are **KeyFactory.getInstance().generatePublic()** and **KeyFactory. getInstance().generatePrivate()**.
5. To encrypt and decrypt with the "RSA" cryptographic algorithm, the class is **Cipher** and the methods are **getInstance()**, **getInstance().init()**, and **getInstance().doFinal()**.

C. "Pemahan Omah @UMB": Server-less 100-Percent-Sevice-Level E-learning System

"Pemahan Omah@UMB" is an e-learning system developed by the author that is free from any Course Management System server (CMS server) dependency. It is a software developed as part of Community Service Activities by Computer Science Faculty of Universitas Mercu Buana in the academic year of 2015/2016 [5]. The software is publicly demonstrated in the academic year of 2016/2017 to a group of high school teachers from South and West Region of Jakarta, Indonesia [6]. Later the software is modified to accommodate the community education process supported by Training and Development Center for Pre-School Education and Community Education of DKI Jakarta Province [7]. The targets of the community service activities are institutions or group of people with inadequate human resource or financial resource to adopt sophisticated information technology infrastructure in an e-learning environment. "Pemahan Omah @UMB" is built, with the hope to support these targets to embrace e-learning environment without the need of any CMS server or the infrastructure to support the server, because they only required providing Android devices and low-bandwidth Internet connection to use the e-learning system [5].

In the application architecture model of Universitas Mercu Buana, e-learning software is one of the activities that supports and manages the data of academic processes, because some of the courses in the curriculum are offered as e-learning courses.

The e-learning software is implemented by paying close attention to the principle of technological independency and ease of use [8]. Currently, Universitas Mercu Buana uses "Moodle" CMS server technology and its infrastructure to server its 30,000 of its students and uses technology independent web browser to access the e-learning content [8]. To serve spesific functions of the e-learning system (such answering quizzes or responding to forums) and to implement the principle of ease of use, the Moodle architecture permits the use of Android-based client software to access the e-learning system through a web service prepared by the Moodle CMS [9]. Most of the parts of "Pemahan Omah @UMB" is inspired by the Moodle CMS and it is developed with the principle of ease of use and the principle of ease of maintenance.

The high availability of the e-learning system is achieved due to the distributed characteristics of the systems. Each software or application that runs on students' or teachers' devices can act as the servers and as the clients at the same time. It means that each installed application can act as an independent distributed server with "minimum" workload. The workload is labeled "minimum" when it is compared to the workload of a CMS server that must deal with hundreds and even thousands of connected clients that must be served in real-time mode. When acting as a server, "Pemahan Omah @UMB" will only have to deal with hundreds of messages, because a teacher is highly probable to have no more than hundreds of students in a specific running semester. All the messages must be dealt only in batch mode, so the system will surely have a very low level of workload. The low level of workload and the batch processing nature of it will ensure that the e-learning system will have a hundred-percent service level [10].

The e-learning software system runs on two groups of application users. The first group is the group who delivers the learning content (teachers) and the second group is the group that is receive the learning content (students). The features for each group is different, but when the software is installed in a user's Android device, at anytime the user can change his/her role from a teacher to a student and vice-versa. With this feature, everyone will need only one complete e-learning software system to fulfill both roles of teaching and learning [5]. The first version of the e-learning software has severe limitations due to the implemented cryptography system and the kind of learning content that it can produce. A detailed discussion of these limitations and the proposed solutions will be postponed until the system requirement specification section in the following text.

III. RESULT AND DISCUSSION

A. System Requirement Specifications

There are some severe limitations in the first version of the e-learning system, so it cannot be called learning technology system architecture or an e-learning system by the definition provided by the IEEE [11]. The system can only support the creation of quizzes and cannot support the creation of learning contents in general. The effort to simplify the business process of exchanging messages between parties through social network applications installed on Android device, also limits the type of content that can be produced by using the e-learning system [5].

With the new requirement so that the new version of the e-learning system can produce and distribute general (not just quizzes) and more complete (can contain both text and image) learning contents, the size of exchanged data grows from the order of hundreds-of-bytes to potentially the order of tens-of-megabytes. If the data is secured by using a cryptographic algorithm and converted into base-64 formatted text, then the result will be a very long and winding text. It is not possible anymore to apply the copy-and-paste scenario, because most popular social network applications only support text message length in the order of tens-of-kilobytes. The proposed solution is to save the long and winding text result into a file inside the Android system storage and then send the file through the "share" mechanism of the Android System. With the "share" to social network application scenario, the extra advantage compared to the copy-and-paste scenario is that less steps are required to distribute learning contents.

Another limitation of the first version of "Pemahan Omah @UMB" is that it uses symmetric cryptographic algorithm to secure data exchange between parties involved in the e-learning process. Due to the nature of this algorithm, the first version of this software has the following restrictions:

1. The security measures to protect the exchanged data is still in the form of "security by obscurity". It means that the measures are not truly an accountable security measures but only a kind of security that only obfuscates and hides the secret data. Therefore, it is still very vulnerable to a determined cracker. A real secure system will not keep unencrypted information in the permanent media storage nor exchange it in an open communication channel.
2. There is no effective way to limit the receiver of data in a group, if there is a need to send the data only to some of the group members. This is a very probable situation in an e-learning setting. For example, a teacher may have the intention to give remedial test to some but not all the students. The symmetric cryptographic algorithm cannot support this process, because there is no effective way to exchange symmetric keys in an open communication channel of the social network media.

There are several software requirements that must be fulfilled if the e-learning system is going to use the asymmetric cryptographic algorithm to secure the data exchange process. These requirements are:

1. Every user of the software must be able to generate public key and private key, since these pair of keys are required for the asymmetric cryptographic algorithm to work.
2. The generated public and private key must be saved in the Android system's storage and the process of saving the private key must be secured with symmetric algorithm cryptography. The extra security measure is required because the private key must be kept secret and must be prevented from falling into any unauthorized person's hand.
3. A teacher must have the public keys of every students in his/her classes, so the teacher can encrypt the data sent to all of them and can prove the identity of the students when they send message to the teacher,
4. All the students must have the public keys of their teachers, so they can prove the identity of the teachers when sending messages to them.
5. Asymmetric cryptographic algorithm uses a great deal of computation resources. Therefore, there is a need to limit the use of the algorithm only in the process of exchanging a small size secret symmetric key. This secret key and the symmetric cryptographic algorithm are what ultimately used to encrypt the exchanged data. It is required that every user of the e-learning software can generate different secret keys each time they want to send message to another group's member.
6. To remove the need to save important information that is not secured with well designed security measures, it is required to provide login-with- password mechanism. With this approach, the software can get the secret key from the password and does not have to save any secret key in the permanent storage. To validate the secret key, the software only needs to keep the hash value of the key.

B. Designing Information Security Mechanism for the New E-learning System

1) Designing Public Key and Private Key Generation Mechanism: One of the requirements of the e-learning system is to generate public-private key pairs. When this pair of key is generated, both of this key must be saved in the permanent storage. Because the public is going to be distributed to senders of messages, the only key that has to be secured before saved is the private key.

As have been mentioned before, the best way to secure data is to prevent saving the data in the permanent storage in the first place. For example, to authenticate its user, software provides a login form and the user can input a password. To validate the password, the software does not compare the password with the originally recorded password, but compares the hash value of the typed password with the hash value of the password accepted during user registration. In the security design applied to the e-learning software, the password is further used as the secret key to encrypt the generated private key. In 128-bit AES symmetric cryptographic algorithm, the key length is 16 bytes/characters [2]. Because the user's password length may be more than 16 characters or less than that, the password is hashed by using the SHA-1 hashing algorithm. The final secret key of the AES algorithm is the first 16 bytes of the produced hash code. Before saving the private key to the permanent storage, it is encrypted with AES cryptographic algorithm, utilizing the 16 characters secret key. Later, the private key can be retrieved from the storage when the user re-enter the password during login session.

The complete algorithm steps in the e-learning software initialization phase, are the followings:

1. Ask user to enter a password.
2. Generate an AES secret key by hashing the entered password with SHA-1 hashing algorithm and by only taking the first 16-bytes of the produced hash value.
3. Generate random public and private keys for RSA asymmetric cryptographic algorithm.
4. Encrypt the private key with AES symmetric cryptographic algorithm, utilizing the AES secret key.
5. Hash the AES secret key with SHA-1 hashing algorithm.
6. Save the public key, the encrypted private key, and the hash value of the AES secret key, to the permanent storage.

2) Designing Student Registration Mechanism: It has been explained before that one of the requirements to implement asymmetric cryptographic algorithm is the sender of the message must know the public key of the receiver and vice versa. Therefore, it is necessary to provide a public key exchange mechanism. In the e-learning software, one of the first exchanged data is the class schedule data sent by teachers to the students. This data is sent to all the students enrolled for the class. In the scenario that uses social network application as the main communication channel, a teacher can create a chat group that represents the class schedule and then invites the entire student enrolled in the class to the chat group. Every time the teacher sends data to this group, for example the class schedule data, each of the group members can import this data into the e-learning software system. If in the future, teachers need to send new data that contains additional quizzes not in the e-learning system yet, they can send the data through this chat group also. The students can then import the additional quizzes data to the e-learning system and work on the quizzes [5].

The proposed algorithm to support the process of exchanging public key in the new version of the e-learning system, involved the addition of registration feature for students, so their public keys can be shared with a teacher.

The complete process of registering a student to a class is described in the following points:

1. A teacher creates a chat group that represents his/her class in a social network application.
2. A student opens the social network application and sends a ping message to the teacher and the teacher replies with a invitation link to join the chat group.
3. The student joins the chat group by selecting its invitation link in the social network application.
4. The student runs the e-learning software and select the registration feature.
5. The e-learning software generates registration data that includes the public key of the student in an encoded text-plain format.
6. The student shares the encoded text to the social network application and send it to the chat group.
7. The teacher copies the encoded registration data to the e-learning software through its class member registration feature.
8. The e-learning software interprets the encoded registration data, saves the student's public key, and adds the student as one of the members of the teacher's class.
9. The teacher shares the class schedule data to the chat group through the schedule sharing feature of the e-learning software. Included in the data is the public key of the teacher. The process of securing the class schedule data with the student's public key, will be discussed in the next section.
10. The student shares the schedule data from the social network application and sends it to the e-learning software that runs in his/her device.
11. The e-learning software imports the schedule data, imports the teacher's public key and saves them to the storage.

3) Designing Security Mechanism for Exchanging Class Data: In the server-free e-learning system, there are two types of messages or data-files sent from the teacher of a class to the students:

1. The first data file contains class schedule data that holds details of the class. The file might also contain learning-content/quiz data and it may include text and images data.
2. The subsequent data-file that is exchanged contains learning-content/quiz data only. This type of data-file may also include text and images data.

The image component of the data is the reason the file size could be in the order of tens-of-megabytes.
There are several choices to design the security measures for the data exchange process:

1. Encrypt each file to each student with asymmetric cryptographic algorithm by utilizing each student's public key.
2. Encrypt each file to each student with symmetric cryptographic algorithm, by utilizing a random secret key for each message. Include in each file, the random secret key that has been encrypted by utilizing each student's public key.
3. Encrypt only one file for all students with symmetric cryptographic algorithm, by utilizing only one random secret key. Continue by encrypting the random secret key with asymmetric cryptographic algorithm, by utilizing each student's public key. After doing this step, there will be several encrypted random secret keys according to the total number of registered students of the class. And then, include every students' identification numbers and each of the encrypted random secret keys in the file.

Because the asymmetric cryptographic algorithm is very computationally expensive, the first choice can be easily discarded as a candidate solution. The second choice is not an efficient solution because a huge file will be produced with this choice, since each student will have a unique encrypted file, so the total size of the produced file, will be approximately the number of students in the class multiplied by the size of the encrypted file. The third choice is the optimum solution, in terms of security measure, computation resource used, and final size of the produced file.

The final algorithm employed by the e-learning system to secure the class data exchanged by teachers and students, can be explained in detail with the following steps:

1. Generate a random 16-bytes AES secret-key.
2. Encrypt the data to be exchanged with AES symmetry algorithm, by utilizing the random secret key.
3. Read all the students' public key from the storage.
4. Encrypt the random secret key with RSA asymmetric cryptographic algorithm, by utilizing each student's public key. After this step, there will be several encrypted random secret keys equal to the number of registered students of the class.

5. Include the symmetrically-encrypted learning-content along with all the students' identification code paired with each of the asymmetrically-encrypted random secret key and along with the teacher's public key, in the final produced file.
6. Share the file to the chat group of the social network application.

The steps that must be fulfilled by the e-learning system, so a student can import the shared class schedule data-file are as follows:

1. Ask student's password in the login form.
2. Generate the hash value of the password with SHA-1 hashing algorithm and then set the AES secret key as the first 16-bytes of the hash value.
3. Read the encrypted student's private key from the storage.
4. Decrypt the student's private key with AES cryptographic algorithm, by utilizing the AES secret key.
5. Find in the shared file, the student's identification code and the student-specific encrypted AES secret key used to secure the shared data.
6. Decrypt the student-specific encrypted AES secret key with RSA cryptographic algorithm, by utilizing the student's private key.
7. Decrypt the encrypted class data with AES cryptographic algorithm, by utilizing the student-specific AES key.
8. Import the class data along with the teacher's public key into the e-learning system.

In step number five (5) of the procedure to import class data by a student, the e-learning system must find the student identification code in the recipient list of the shared file, so it can find the student-specific encrypted AES key. This step provides an opportunity to add another important feature to the e-learning system, which is the feature to limit the receiver of the shared file only to a specific person or specific group of persons.

The e-learning system can achieve this end by providing a form to let the teacher changes the status of each student as an active or as a non-active recipient of the teachers' shared file. When the e-learning system wants to create the recipient list and embed the student-specific encrypted AES keys in the shared file, it will first check whether the status of a specific student is active or non-active before putting the student in the list. A non-active student will not have his/her encrypted AES key in the shared file, so it is not possible for the non-active student to receive the encrypted content contained in the shared file. With this feature, the teacher does not have to deliver the file individually to each allowed student, but it is enough to deliver one file to the chat group and then the e-learning system will determine if the student that is trying to import the content, is in the recipient list or not.

IV. CONCLUSION

This paper describes the successful implementation of a business process or procedure to use RSA asymmetric cryptographic algorithm and AES symmetric cryptographic algorithm to secure a file attachment that is potentially very large in size, when it is sent through a social network application. The procedure starts with providing a class registration feature that lets students to register their public key to a teacher's class. The next step in the procedure is encrypting the file attachment by the teacher's e-learning software, by using AES symmetric cryptographic algorithm and utilizing a random secret key. Then, the random secret key is encrypted with RSA asymmetric cryptographic algorithm, utilizing each student's public key. After this process, each student will have his/her specific encrypted secret AES key. Then the student's e-learning software can import and decrypt the file attachment for each student by first decrypting the AES secret with RSA asymmetric cryptographic algorithm and utilizing each student's private key. The last step in the process of importing the attached file is decrypting the file with AES symmetric algorithm and utilizing the student-specific AES secret key. All the implemented RSA asymmetric cryptography and AES symmetric cryptography utilize the "javax.crypto" library of the Java development environment in the Android system. All the related classes and methods can be implemented successfully and without any problem during the new features development of the e-learning system.

REFERENCES

1. L.L. Peterson & B.S. Davie (2000). Computer Networks: A Systems Approach, 2nd ed. San Diego, USA: Academic Press.
2. J.E. Goldman & P.T. Rawles (2004). Applied Data Communications: A Business-Oriented Approach. New Jersey, USA: John Wiley & Sons, Inc.
3. Nurhaida, I., Ramayanti, D., & Riesaputra, R. (2017). DIGITAL SIGNATURE & ENCRYPTION IMPLEMENTATION FOR INCREASING AUTHENTICATION, INTEGRITY, SECURITY AND DATA NON-REPUDIATION. IRJCS:: International Research Journal of Computer Science, Volume IV, 04-14. doi: 10.26562/IRJCS.2017.NVCS10080.
4. (2017) "Classes and Methods of javax.crypto Package". [Online]. Available: https://developer.android.com/reference/javax/crypto/package-summary.html

5. Anis Cherid, Ida Nurhaida & Bambang Hariyanto (2015). "Sosialisasi Perangkat Lunak Pengelolaan Kelas Elearning Tanpa Server CMS Berbasis Perangkat Telepon Bergerak kepada Guru SMA di Jakarta Barat," Universitas Mercu Buana, Jakarta, Indonesia. Community Service Activity Technical Report.
6. Anis Cherid, Ida Nurhaida & Bambang Hariyanto (2016). "Sosialisasi Perangkat Lunak Pengelolaan Kelas Elearning Tanpa Server CMS Berbasis Perangkat Telepon Bergerak kepada Guru SMA di Jakarta Barat - Tahap 2," Universitas Mercu Buana, Jakarta, Indonesia, Community Service Activity Technical Report.
7. Anis Cherid, Diky Firdaus & Andi Nugroho (2017). "Implementasi Model dan Materi Pembelajaran E-Learning Berbasis Android tanpa Server Web pada Lembaga Pendidikan Non-Formal Provinsi DKI Jakarta," Universitas Mercu Buana, Jakarta, Indonesia, Community Service Activity Technical Report.
8. Harwikarya & Raka Yusuf (2016). "Enterprise Architecture Development of Mercu Buana University Based on TOGAF: An Introduction," IRJCS:: International Research Journal of Computer Science, Volume III, Issue 09.
9. J. Piguillem, M. Alier, M.J. José Casany, E. Mayol, N. Galanis, F.J. García-Peñalvo, & M.Á. Conde (2012). "Moodbile: A Moodle Web Services Extension for Mobile Applications," in Proc. Moodle ResearchConference, 1st, p. 148-156, Heraklion, Crete-Greece.
10. Anis Cherid (2017). "Rancang Bangun Perangkat Lunak Pengelolaan Kelas E-Learning tanpa Server CMS yang Berbasis Android," Format Journal, Volume 2, Issue 01.
11. Learning Technology Systems Architecture LTSA (2003), IEEE Std 1484.1.

