2011 International Conference on Electrical Engineering and Informatics
17-19 July 2011, Bandung, Indonesia

Text and File Encryption Application for BlackBerry Using Cipher Feedback 8-bit Mode

Matthew Wangsadiredja[1], Dr. Ir. Rinaldi Munir, M.T.[2]

Informatics Engineering Study Program, Bandung Institute of Technology,
Ganesha Street 10, Bandung – 40235, Indonesia
[1]matt_jcs@yahoo.com
[2]rinaldi@informatika.org

Abstract— **BlackBerry is not just an ordinary smartphone which rich in its multimedia features. BlackBerry also has very reliable internet services, include BlackBerry Messenger, e-mail, other instant messaging, and other internet service that Blackberry Internet Service (BIS) covers. However, there's an open question that all services that go through BIS is safe or not; considering the data includes text and files which is sent through BIS have to pass through the main server which is located in Canada. Therefor in this paper we chose to make software that can encrypt and decrypt text and even files, which will be sent through BlackBerry's internet or just to be stored in the BlackBerry device.**

In this paper, we use cipher block encryption algorithm with Cipher Feedback (CFB) 8-bit mode. This mode is chosen because it suits with the data inside BlackBerry device which is 8-bit aligned. The encryption algorithm uses cipher block encryption with 64-bit block size. We designed the algorithm based on Feistel network combined with some encryption/decryption algorithm. Then the cipher block encryption algorithm is implemented in software based on Java for BlackBerry. In this software development, we use the IDE Eclipse Ganymede with certain plugins.

The expected result is software for BlackBerry capable to encrypt and decrypt some texts, either directly typed or saved as files on the BlackBerry device. This software also handles specific file types, such as picture and audio by preserving their file header, and also handles text encryption by transforming the ciphertext data to hex digits. After conducting some experiments, we conclude that the software has good compatibility with many kinds of BlackBerry devices and has good security level.

Keywords— **BlackBerry, cipher block encryption, cipher feedback 8-bit.**

I. INTRODUCTION

The most interesting feature from BlackBerry is its exclusive internet services. RIM is using its own server for all internet based services, making better access to the internet. In short, all activity using BlackBerry services all over the world is centralized on RIM server on Canada. This is different from internet services using other devices, which are using server in each country. BlackBerry network schema is shown in Fig. 1, which is always centralized in Canada and making each network provider to have private lane from each country to Canada.

On the contrary, this condition is assumed as a potential risk by some countries such as UAE and Saudi as in [1], because all BlackBerry user data is stored on the server which is located in Canada, which mean data is located in outside of local jurisdiction. Hence there are some country requested RIM to build server on their countries. Unfortunately, RIM has not yet granted any requests and still centralizes the data in Canada as in [1].

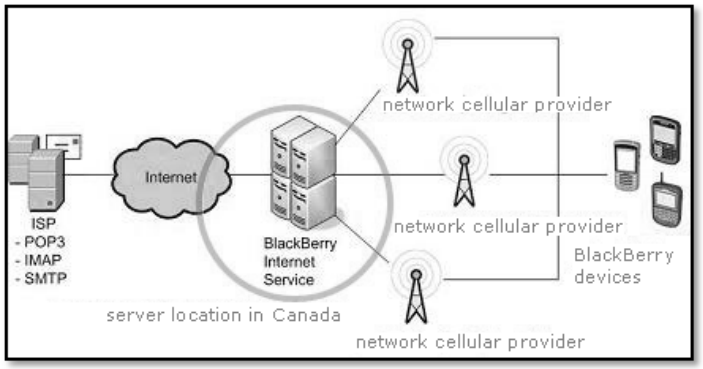

Fig. 1 BlackBerry Internet Service network diagram as in [2]

Most of users choose BlackBerry because BlackBerry has good instant messaging and reliable e-mail features. BlackBerry also has an exclusive instant messaging service, BlackBerry Messenger or BBM. This application facilitates the users to send texts and files via this messenger. The BlackBerry users in Indonesia are about 4.5 million in 2010 as in [3] and the number is still growing rapidly. This point becomes notice by some countries government including Indonesia. There are so many data, even texts or binary files such as picture, audio, and other document files which are stored in RIM's server in Canada when people communicate using BlackBerry services. The most important question from this situation is whether the data is really keep private by RIM, or even misused by unwanted side.

Nowadays we need a media to help people who really want their data safe when they want to send it or just keep it inside the device. Furthermore we need a media to convince the government that all important data transaction within BlackBerry user is keep in safe, even the data center is located in Canada.

978-1-4577-0751-3/11/$26.00 ©2011 IEEE

Based on those problems, we chose cryptography on BlackBerry topic for this paper. As in [4], cryptography is "art and science to keep message secure". Then we make the practical solution by making software that capable to secure BlackBerry users' data. The software will be based on cryptography field using cipher block method. Cipher block algorithm is an encryption algorithm which split plaintext data bit to same size then encrypting the data block by block. To make the process more efficient, we chose Cipher Feedback (CFB) mode, which can encrypt data in smaller unit like stream cipher but with cipher block security strength.

II. CFB 8-BIT ALGORITHM

On Cipher Feedback mode, data is encrypted to smaller unit then the block size. By using this mode certain bit size can be encrypted exactly, for example 1-bit or 1 character (1-byte) as mentioned in [5]. The schema on Fig. 2 from [6] shows an example of CFB algorithm encryption that use 64-bit block size and using 8-bit (1-byte) mode.

Fig. 2 shows that a single byte can be encrypted or decrypted exactly on size using cipher block 64-bit with CFB mode. First, the shift register is initialized by the initialization vector (IV), then using the encryption algorithm producing 64-bit data output. As in [5], IV is a binary vector for initialize the input block for CFB or OFB mode as a random value block. After get the encrypted 64-bit data, the left-most byte (8-bit from the left) is taken and processed with XOR bit operator with the plaintext data. This 8-bit XOR result then stored as ciphertext and using the same 8-bit result to shift the input block data. This encryption algorithm is repeated until all the plaintext has been encrypted.

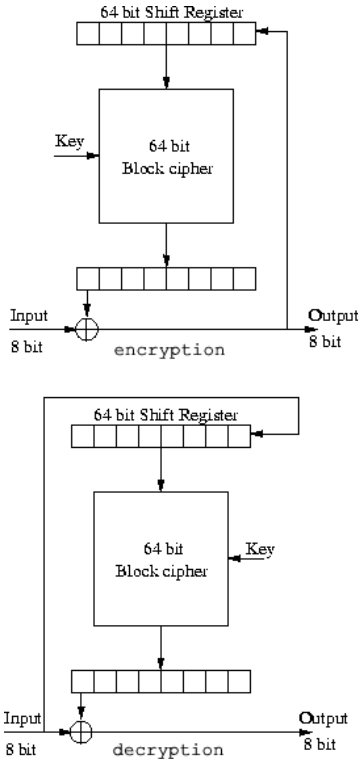

Fig. 2 Encryption and decryption using CFB 8-bit mode with 64-bit block size

The initialization vector or IV we use here has requisite, that the IV value is not private, but has to be different for each encryption with the same key. The purpose of using IV is it will make every encryption process has different pattern even if using the same key. Every bit errors in ciphertext will make the correspondent plaintext block error at same position. That will make condition, when using CFB 8-bit mode and have some bit error at the ciphertext block, all plaintext in correspondent block size (e.g. 64-bit block) will be corrupted, meanwhile the other block still can be decrypted as usual.

To explain the CFB 8-bit we use in this paper more detail, see the schema in Fig. 3 below.

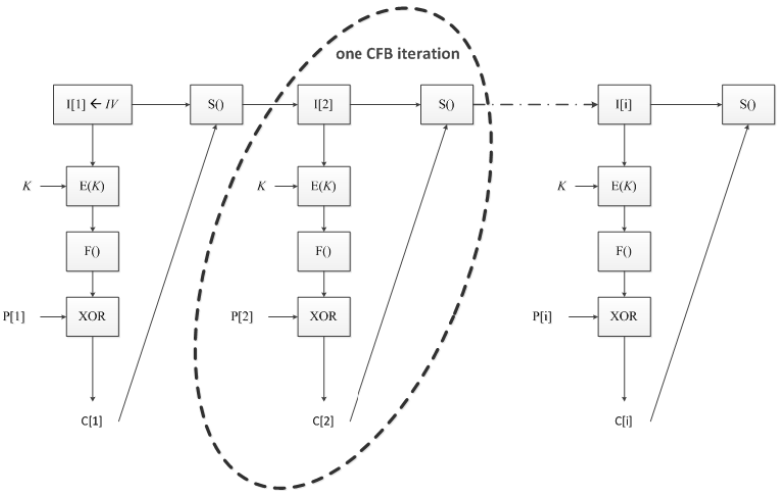

Fig. 3 Step by step encryption using CFB 8-bit with 64-bit block size

The following below is the explanation for notation used in Fig. 3:

IV: Initialization vector, always random and different bit value for CFB encryption process. IV only used once, it is to initialize the first iteration of input block. IV has 64-bit data length.

I[i]: Input block, is going to be encrypted by the E function. I[i] has 64-bit data length. Only for the first iteration, the input block value is taken from IV, for the next iteration the input block is taken from S() function.

E(K): The encryption function, which encrypts the 64-bit data block using the key K. The key K is entered manually by the user.

F(): Filter function, a function to filter left-most byte (8-bit from the left from the result of encrypted block.

P[i]: Plaintext data block, it will be XOR-ed with the result of filter function. For each iteration, the plaintext use 8-bit data length.

C[i]: Ciphertext block, first store ciphertext block from all iteration of the CFB, after all the iteration finish, all ciphertext

block is merged into whole ciphertext data. Each block has 8-bit data length.

S(): Shifting function, a function to shift the first 8-bit of the previous input block by inserting the 8-bit current ciphertext data block into the back part of the input block. The result of this shifting function is used of the next iteration's input block.

i: iteration count for this CFB mode. The iteration count for CFB 8-bit mode is from 1 to plaintext data size in bit divided by 8 bit.

III. Encryption Algorithm of E(K) with 128-bit of Key

CFB mode can be used with many kind of encryption algorithm. For example CFB is used with the DES (Data Encryption Standard) as in [5]. DES use 128-bit data length for the key, which is processed by key generator before the main algorithm process. In this paper, we use our own custom design algorithm. The encryption algorithm is based on Feistel network schema and has some binary manipulation process which is iterated for four times. The key we use here is 128-bit data length and will be separated into four parts; each part is used for the iteration of the Feistel method. The E(K) algorithm flow is shown in Fig.4 below.

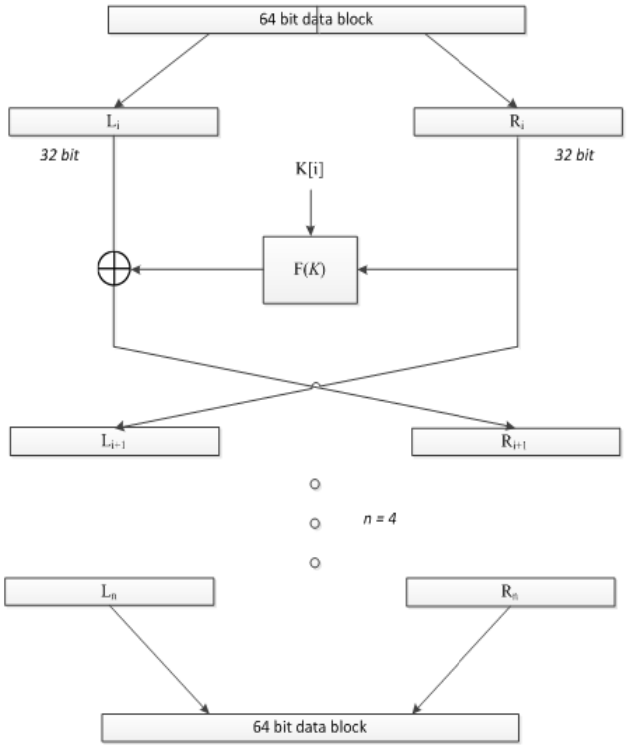

Fig. 4 Algorithm E(K) base on Feistel network with 64-bit block size

First, the 64-bit data block is divided into two parts, left block (L) and right block (R), each with same size of 32-bit. The R block is stored for the next L block, meanwhile the next R block is result from previous R block which is processed through the F function using K key, then XOR-ed with the L block. This process is repeated by four times, each iteration using different key. The key is generated from the 128-bit key data, then divided into four part as shown on Fig. 5 below.

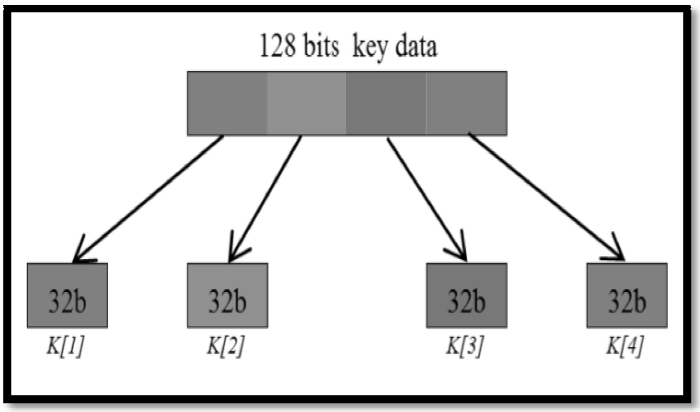

Fig. 5 Key generator process, from 128-bit into four block of 32-bit key data

After finished with generating process, the key is used in F function inside the Feistel network schema. F function works in 32-bit data block, the input is R block. This function will have three phases as shown on Fig. 6 below:

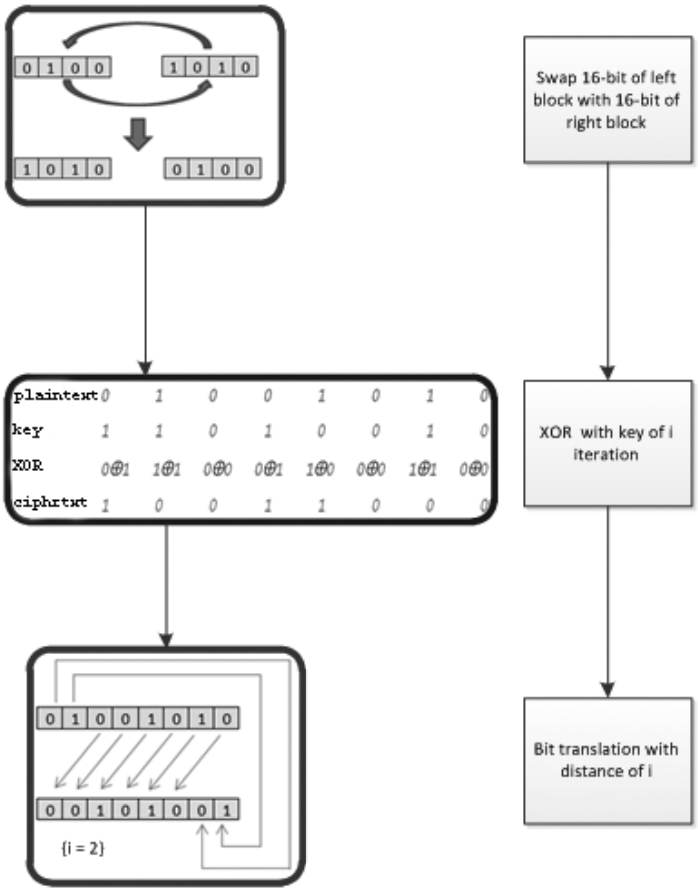

Fig. 6 Three phases of F function

The following below is the explanation for step by step in Fig. 6:

1. Swap function, swap 16-bit of left block with 16-bit of right block.
2. XOR with key of i iteration, i value is depend on the iteration count. On first iteration, XOR process uses key from first block, and so on for the next iteration until the fourth one.
3. Bit translation with distance of i, moving the bit by i-bit, making each iteration has different bit moving distance.

IV. CFB 8-BIT MODE ALGORITHM IMPLEMENTATION ON BLACKBERRY APPLICATION

The software has capability to encrypt and decrypt both text and binary file with private key entered by the user. This software is an application which will run on BlackBerry devices with operation version greater than 4.6.0 because we use BlackBerry API version 4.6 as in [7]. The application encrypt the data with CFB 8-bit mode, and the block encryption use the E(K) algorithm which is previous mentioned on chapter III. The general schema for the software is mentioned in Fig. 7 below.

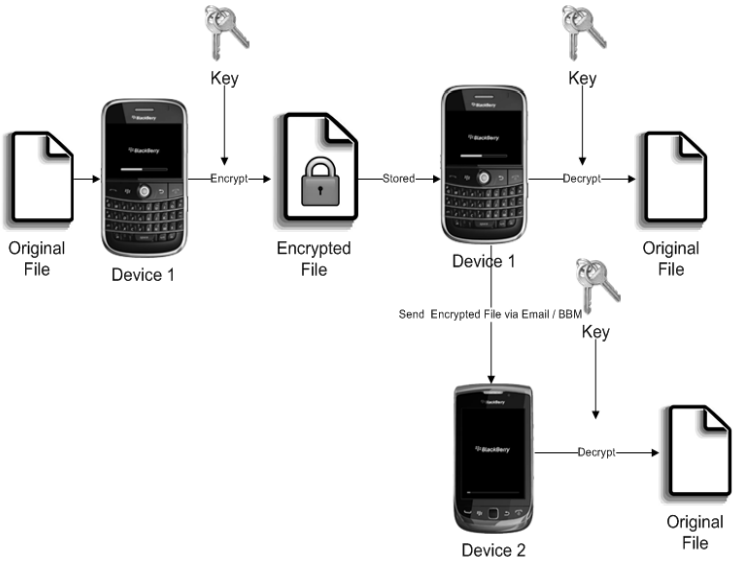

Fig. 7 General software schema

A. Functional and Non-functional Requirement

Functional requirement for the software are:

1. Capability to encrypt and decrypt both text user typed and file inside BlackBerry devices using the key input by the user.
2. Have compatibility with all BlackBerry devices with operation system greater than 4.6.0.
3. Preserving file header of some kind of file from BlackBerry, such as audio and picture.

Non-functional requirement for the software are:

1. Providing high security application, so the encrypted text or file is guarantee secured.
2. Process the encryption and decryption in high speed.
3. Provide application with good user interface, i.e. easy to understand and use by every BlackBerry user.
4. Providing online link, so the application can be easily downloaded.

B. Class Implementation

The software is developed using Java for BlackBerry language with IDE Eclipse Ganymede. All classes which have been implemented are shown on Table I below.

TABLE I
CLASS IMPLEMENTATION

No	Package	Class
1	crypto	BitSetHelper
2	crypto	CFB
3	crypto	Feistel
4	filepicker	FilePicker
5	filepicker	FilePickerDirListField
6	filepicker	FilePickerFileHolder
7	filepicker	FilePickerFileListField
8	filepicker	FilePickerListener
9	helper	Email
10	helper	HexByte
11	pv	HomeScreen
12	pv	Pv
13	component	Colors
14	component	CustomButton
15	component	CustomEditField
16	component	CustomePasswordField
17	component	FileIO
18	component	Fonts
19	component	ImageField
20	component	TitleField

C. Application Interface Implementation

The application interface is focused on a single main page. In this page, there are nine fields as shown on Fig. 8 below:

Fig. 8 Application main page appearance

1. Application header for the title
2. Choose file button, will invoke a file picker if pressed
3. Text field showing the file path that has been taken by the file picker
4. Text field to write some text to encrypt or the ciphertext to decrypt
5. Text field to write the key that will be used for encryption and decryption process
6. "Encrypt!" button, the button to start the encryption process after input the key and choose a file or write some text.
7. "Decrypt!" button, the button to start the decryption process after input the key and choose a ciphertext file or write ciphertext hex digit.
8. Picture field that will show picture from the selected file if any.
9. Text field for the result of the decryption or encryption process of text.

V. SOFTWARE TESTING

A. Test Goals

The goals of this testing section are to answer the functional and non-functional requirement which has been mentioned before on chapter IV. The goals are:

1. Test application compatibility in different BlackBerry devices.
2. Test software handling while encrypting some known file types.
3. Test the correctness of encryption and decryption process.
4. Test the security level by three main security goals for application; confidentiality, integrity, and availability.
5. Show step by step to download the application.

B. Test Result Analysis

Test Case-1

On this test case, we try to check the compatibility by installing the application on different BlackBerry devices. After that we inspect the compatibility and the interface. Screen resolution and operating system which are used on this test is shown in Table II.

TABLE II
USED DEVICES ON COMPATIBILITY TEST

No	Device	Type	Resolution	Operating System
1	*BlackBerry 9000 Bold*	Simulator	480 x 320 *pixel*	v4.6.0.92
2	*BlackBerry 9800 Torch*	Simulator	360 x 480 *pixel*	v6.0.0.141
3	*BlackBerry 8520 Gemini*	Device	320 x 240 *pixel*	v4.6.1.314
4	*BlackBerry 9700 Onyx*	Device	480 x 360 *pixel*	v5.0.0.405

From this test, we conclude that the software compatibility is good, because it can be operated on some different BlackBerry operating system, e.g. 4.6, 5.0, and 6.0. And the software has proper interface too because objectively the interface is suit some BlackBerry devices with different resolution.

Test Case-2

To test the validity of preserving file header, we need to encrypt the files first. Files we used are "audio.amr" and "image.jpg". After encrypt the files with key "1234", the ciphertext file names are "audio_CPR.amr" and "image_CPR.jpg". Then we open the files using BlackBerry media explorer, and both the files still known as audio and image files. The audio ciphertext file is shown on Fig. 9, and the image ciphertext file is shown on Fig. 10.

Fig. 9 Audio ciphertext opened by BlackBerry media explorer

Fig. 10 Image ciphertext opened by BlackBerry media explorer

Test Case-3

To show the correctness of the process, we check using real BlackBerry devices, they are BlackBerry 8520 and BlackBerry 9700. From this test, we concluded that the encryption and decryption by the software is correct. It is because the real plaintext only can be retrieved if decrypting with the same key when encrypting it.

Test Case-4

Confidentiality: test is done by using debug mode on the IDE Eclipse Ganymede, and then checked some important variables. This debug mode is equal as unwanted software that wants to steal the value such as key or plaintext from our software. The software passed this test, because important data cannot be taken by other software without any permission.

Integrity: test is done by manipulating plaintext, ciphertext, and even the key which are used in encryption and decryption process. This test is done by encrypting and decrypting the text:

```
Ini adalah pesan rahasia.
```

Using key:

```
12345678
```

First, we test by encrypt with same variable three times, the result in hex digits are:

1st encryption

```
401EFA37C93BC0DA6C048E7B55D2286F116BDBF421E71D7E35
E073E9322B9A6F54
```

2nd encryption

```
2472779C27B554ED2799143AA0FA192877AE5F76F4532196EC
8BC5B03CF1A3055C
```

3rd encryption

```
69EBE95F311E2282C940BD177C58168E916209BAFF14B57333
99D2A8B94467430B
```

Although using same plaintext and key, the ciphertext always different because we use the IV. IV always has different value in every encryption, so that the ciphertext value always different too. Then we try to test the decryption process. We manipulate the ciphertext and key then compare the decryption result as in Table III.

TABLE III
DECRYPTION MANIPULATION FOR INTEGRITY TEST

Ciphertext	Key	Decryption Result
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	12345 678	Ini adalah pesan rahasia.
301EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	12345 678	9i adaah pesan rahasia.
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F55	12345 678	HNi ad`ah pesan rahasia.
401EFA37C939C0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	12345 678	Ini afc,ah pg³an rahasia.
101EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	12345 678	°Bs 9¼½ì Y¼ =¬-©
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F5	12345 678	_Ò³Ìah pesan rahasia
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	02345 678	Z}z3rwrr{3cv`r} 3ar{r`zr=
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	22345 678	òÕÒÚßÚ×ÚÓËÞÈÚÕÉ ÚÓÚÈÒÚ
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	12345 678	·°ù¸½¸µ¸±ù©¼ª¸· ù«¸±¸ª°¸÷
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	01234 5678	4]]]S
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	12345 6789	Åâå¬íèíàíä¬üéÿí â¬þíäíÿåí¢
401EFA37C93BC0DA6C048E 7B55D2286F116BDBF421E7 1D7E35E073E9322B9A6F54	01234 56789	ªÃÃÃÍ

Availability: test is done by trying some exception such as wrong input of ciphertext format, empty key, empty file, and some other exceptions. From the test, we concluded that this software also passed availability test well.

Test Case-5

The security application needs to be easily downloaded, because we need feedback for improving the security. We have uploaded the executable file on Getjar.com. To download the application simply direct the browser to http://getjar.com/pv from any BlackBerry devices, then the site will give direction to download the application as shown on Fig. 11. From this test, it is concluded that the software is easy to install and well distributed, so we expected there are many users send some feedback to improve the software security and performance.

Fig. 11 Downloading application progress

VI. CONCLUSIONS

Generally, the Cipher Feedback 8-bit algorithm is well implemented. From the testing we know that the application has capability to run in various kinds of BlackBerry devices. The encryption and decryption process is well functioned too, the decryption process only can be executed by the correct key. File header successfully preserved for audio and image files, so the ciphertext files still known as the original file. The application has good security level, which has passed some security test such as confidentiality test, integrity test, and availability test.

REFERENCES

[1] (2010) CNN Wire Staff. [Online]. Available: http://edition.cnn.com/2010/WORLD/meast/08/03/saudi.arabia.*BlackBerry*

[2] *BlackBerry* Internet Service 3.0 Up And Running For North America. (2010) BlackBerryrocks homepage. [Online]. Available: http://BlackBerryrocks.com/2010/03/29/BlackBerry-internet-service-3-0-running-north-america-features-enhancements-news/

[3] (2010) Kompas Tekno. [Online]. Available: http://tekno.kompas.com/read/2010/10/25/16285320/Data.Center.BlackBerry.Harus.Dibangun.di.Indonesia-12

[4] Schneier, Bruce. *Applied Cryptography 2nd*. John wiley & Sons. 1996.

[5] Federal Information Processing Standards Publication 81.(1980) *Announcing the* ***Standard for DES MODES OF OPERATION.*** [Online]. Available: http://www.itl.nist.gov/fipspubs/fip81.htm

[6] Munir, Rinaldi. "*Diktat Kuliah IF5054*". Informatics Engineering Department, Bandung Institute of Technology: 2005.

[7] *BlackBerry* Java Development Environment version 4.6.0. Development Guide. Canada: Research In Motion Limited: December 5, 2009.