Comparison of cryptographic methods Triple DES, AES and a method based on the arithmetic of elliptic curves (ECC) on the Android mobile platform

Milan Oulehla, David Malaník

***Abstract*—**The paper presents a comparison of three cryptographic methods on mobile devices with the Android operating system. Specifically, methods compared were these: cryptographic method based on the arithmetic of elliptic curves, Triples DES and AES. They were chosen because they seemed to be most appropriate due to the limited processing power of today's mobile devices. For comparison purposes of cryptographic methods, a special cryptographic benchmark was created on the Android platform. Overall, 6660 test files were encrypted. The testing was performed on actual hardware, because tests carried out on an emulator might be misleading. The results were then processed using regression analysis. The tests have shown that the most appropriate cryptographic method for the Android platform is AES with 256-bit key.

***Keywords*—**arithmetic of elliptic curves, Android, cryptographic benchmark, comparison, symmetric cryptography, triple DES.

I. Introduction

THE trend in the last five years is the massive advent of smartphones and tablets with Android OS in both corporate and private sector.

Android was created in 2003 as a global, open operating system for the wireless future. It is a free, open source mobile platform that any coder could write for and any handset maker could install. It is based on a modified version of the Linux kernel. Google Inc. bought Android in July 2005 to take advantage of the seamless Web access; it was introduced as applicable mobile platform to consumers in late 2008. An application program ("app") called "Market" is preinstalled on most Android devices and allows users to browse and download apps published by third-party developers, hosted on Android Market. This system is currently the fastest developing mobile platform [1].

Data of personal and corporate character is increasingly stored in mobile devices. Private and public institutions have their data available in electronic form in order to generate profits and provide services and information. Data belongs to intangible assets that create the value and know-how of organizations. Theft of business contacts, accounting data or manufacturing processes can cause major financial problems of affected organizations. For example, theft of Samsung Galaxy Tab® S 10.5" 16GB, Dazzling [2] with plans of new products will not cause a loss of only $500 for the hardware. More significant damage would occur if the plans were acquired by competition. In some cases, the damage may exceed millions of dollars. Unfortunately, there is currently a disparity between the value of data stored in corporate mobile devices and their protection. As one of the most suitable means of logical data protection is cryptography [3]. Not only does it protect data in the event of physical loss or theft but it also protects the device from "internal" losses caused by malware which users installed from Google Play [4]. In this case, such applications request permissions which are not necessary for their functionality. Users also install applications from unknown sources such as web sites. Based on the data from "Roll call release for police, fire, ems, and security personnel," issued July 23, 2013, it is clear that these risks should not be underestimated.

Android is the world's most widely used mobile operating system (OS) and continues to be a primary target for malware attacks due to its market share and open source architecture... [5]. Next, it states: Tricks users into installing malicious applications that enable malicious actors to steal sensitive information, including financial data and log-in credentials [5]. The entire research was carried out on the Android platform.

II. Choice of Cryptographic Methods

Cryptography on the mobile platform has its pitfalls. The cryptographic methods have relatively high computing requirements and at the same time, the performance of current mobile processors in the main user segment is limited.

Milan Oulehla is with the Faculty of Applied Informatics, Tomas Bata University in Zlín, Nad Stráněmi 4511, 760 05 Zlín, Czech Republic (corresponding author to provide phone: +420-732118306; e-mail: oulehla@fai.utb.cz).

David Malaník is with the the Faculty of Applied Informatics, Tomas Bata University in Zlín, Nad Stráněmi 4511, 760 05 Zlín, Czech Republic (e-mail: dmalanik@fai.utb.cz).

The work was performed with financial support of research project NPU I No. MSMT-7778/2014 by the Ministry of Education of the Czech Republic and also by the European Regional Development Fund under the Project CEBIA-Tech No. CZ.1.05/2.1.00/03.0089.

Another important factor that should be mentioned is the relatively small battery capacity of mobile devices. This fact means that encryption methods, which we consider to be secure, are not applicable for a permanent transparent encryption of entire persistent memories of mobile devices in real time. Therefore, great attention was paid to the selection of suitable cryptographic methods

The cryptographic method based on the arithmetic of elliptic curves [6] was chosen as first. It is known to provide better security per bit than RSA; at the same time it can be feasibly implemented on embedded systems at higher speeds and less memory requirements [7]. Next, the AES (Advanced Encryption Standard) [8] method was chosen. In 2003, the American government issued a declaration which stated that the AES method may be used for protection of classified information [9]. The third method was chosen TripleDES [10]. It is the successor to the DES method which was developed in 1970s. DES has good computing speed but it is no longer considered to be safe because it uses only 64-bit key (56 effective bits). TripleDES uses DES three times, which increases the key length to 168 bits (112 effective bits). Triple DES should have a good balance between the encryption speed and relative safety. Both AES and Triple DES are block ciphers. That means they do not encrypt all bytes of the file at once but in smaller sections called blocks. These blocks have a fixed size. AES and DES have different block sizes. AES works on 16-byte blocks and DES on 8-byte blocks. It is very rare that the encrypted file has size equal to n multiple of the size of the block so it is necessary to perform padding. The issue of padding on the Android platform is dealt with in [11].

The operating system Android has specific management of RAM memory so there are two file sizes: Heap ready size - File size which is allowed to be encrypted in RAM all at once. Heap not ready size - File size which is not allowed to be encrypted in RAM all at once. These files must use a buffer for encryption. And they must be encrypted part by part.

The performance of the cryptographic methods was tested on heap ready files. The measured data was subsequently statistically processed.

III. Creation of Cryptographic Tests and Data Selection

We created cryptographic tests with emphasis on real conditions. Thus we selected these types of files:

- *.mp3 files - Users can record their voice notes. Audio notes can be recorded for example by "Hi-Q MP3 Voice Recorder" application. [12]
- *.jpg files - Users can take pictures with their smartphones. Android has native support for taking photographs.
- *.mp4 files - Video files can be recorded by mobile device. Android has native support for video recording.
- *.txt files - Short text notes can be made by using many Android applications, e.g. Simple Notepad [13].

All of these files can contain sensitive information and users may wish to encrypt them. Android mobile devices are often used for viewing the content. Files which a user creates in mobile devices are very small. The files typically have size about several megabytes. This fact was reflected in the structure of the test files:

- files up to 250 KB.
- files up to 500 KB.
- files up to 1 MB.
- files up to 3 MB.
- files up to 5 MB.

The most of Android devices are able to encrypt at once files which have this size by using method doFinal. Method doFinal is provided by class Cipher (Encrypts data in a single-part operation [14]). The result of encryption can be written at once to a file on a SD card or to internal, persistent memory of a mobile device.

Since the encryption using the doFinal method is relatively quick, it is practically usable even on inexpensive or older devices. For example: LG P500 Optimus One which has CPU ARM 11 performs AES encryption of 1 058 496 B file with 256-bit key in 1.72 s. The time includes both encryption and writing of encrypted file to a SD card. This CPU works on low frequency 600 MHz [15]. Data encryption in a single-part operation is fast but it also has some risks. The main risk is OutOfMemoryError. There is a danger when the Java Virtual Machine cannot allocate an object because it is out of memory, and no more memory could be made available by the garbage collector [16]. And the exact heap size limit varies between devices based on howmuch RAM the device has available overall. If your app has reached the heap capacity and tries to allocate more memory, it will receive an OutOfMemoryError [17].

For example, if we try to encrypt a file of 10 583 904 bytes by using phone LG P500 Optimus One and doFinal method, the Android operating system terminates the encryption. After then we can read the following message in logcat: “dalvikvm-heap： Out of memory on a 10583936-byte allocation“. On the other hand, the encryption of the same file on your Samsung i9505 Galaxy S4 will be fine. This means that the application programmer who wants to use for encryption fast doFinal method must first call getMemoryClass () to determine whether the application has enough space in memory for the encryption of a particular file. The encryption application needs space in memory for:

- unencrypted file
- encrypted file
- all other objects which an application needs for work.

If there is not enough space in memory, it must be done by using Buffer encryption. In this case instances of Cipher class have to use some variant of the update method. The encryption using the update method is slower, but there is no OutOfMemoryError. The size of encrypted files is not limited by the size of available space in RAM which the system is able to allocate to the application. Using update method has

some problems in Android operating system. For more information [11].

The differences between 'At once encryption' (As it was written above, it uses doFinal method.) and 'Buffer encryption' are shown in Figure 1.

The encryption methods which were used for testing:

- Triple DES (DESede/ECB/PKCS5Padding), provider: AndroidOpenSSL, length of the key: 168 bits. Note: DESede is alias of TripleDES method. This alias is usually used with Cipher class.
- AES (AES/ECB/PKCS5Padding), provider: AndroidOpenSSL, length of the key: 256 bits.
- ECC (ECIES, FlexiEC), provider: FlexiProvider, 160-bit elliptic curve.

Isolated observation of encryption process running in the RAM does not have relevant results. We need more complex view for detection of the specific character of the encryption on the Android platform. Comprehensive encryption process is a sequence of all operations that end by storing encrypted file on a user device. Sequence of the operations is following:

- loading data from a file stored on the SD card or in persistent memory to the operating memory,
- creation of a randomly generated key,
- encryption of a file by using a randomly generated key.

If we want real-life results of measuring, none of these steps can be omitted. Same conditions for all tests were ensured by using same microSD card for all tested device. Passwords were not used for the encryption but we used randomly generated encryption keys (as mentioned earlier). Key lengths were chosen by the used method, safety and time demands. Some Android mobile devices have no SD card slot, for example Nexus 5. Therefore, encryption tests were performed both on the SD card and in internal memory of the mobile device.

IV. Units Creation of a Cryptographic Benchmark

The principle of measuring the performance of the application:

- The application stores the start time.
- The application starts the code whose performance is measured.
- The application stores the end time.
- The application calculates the elapsed time. Elapsed time is the difference between end time and start time.

Although the principle of measuring the performance of the application is very simple, its practical implementation is very difficult. There are many pitfalls that can distort the results of measurement. Correct cryptographic benchmark has to avoid them. The following is the list of the biggest problems and actions which were done to prevent the distortion problems:

No. 1: The measurement is not performed in a separate thread. If the encryption task shares the main thread with operation of GUI, it may cause that the measured time will be extended by time needed for the GUI operations. Also, the time may be extended by time of other operations from the main thread.

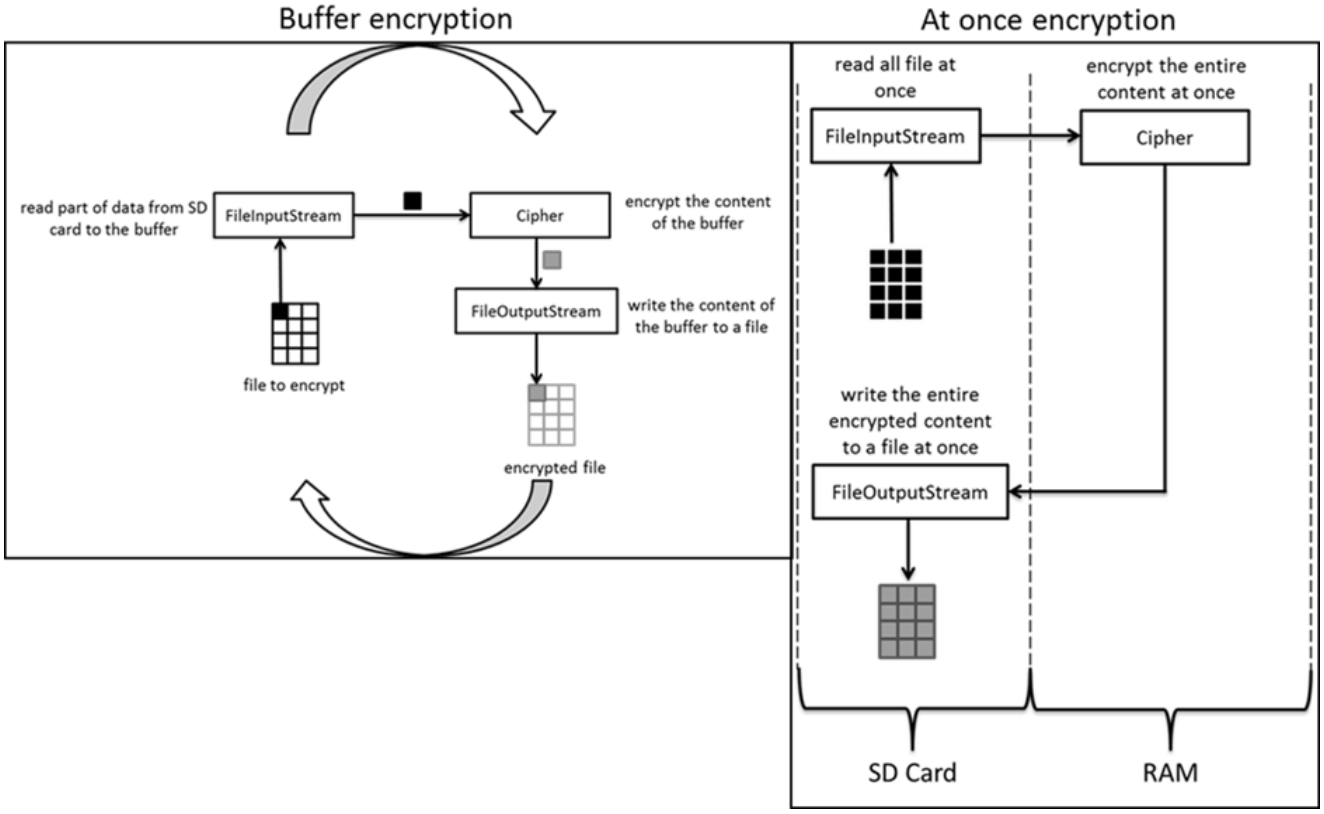

Fig. 1. The differences between At once encryption and Buffer encryption

These operations from the main thread have nothing to do with the test encryption method and they can distort the results of measurement too.

Solution of the problem No. 1: The application will create a special thread only for encryption testing. This thread will create instance of cipher class of a particular encryption method. Then, constructor of the cipher class will create everything what is needed for the encryption. The start time will be saved and testing thread will call encrypt () method. Encrypt() will perform encryption. After the encryption is finished, the end time will be saved. In the end, the elapsed time as difference between end time and start time will be calculated.

Problem No. 2: One file is encrypted with a particular cryptographic method only once. During the first encryption a lot of activities are performed. But those activities aren't directly connected with the encryption. It is more about the behavior of the JVM. Thus, it needs more iteration. It is called as a warmup code.

Solution of problem No. 2: The tests which were conducted in our research showed that the first iteration almost always lasted longer than the other iterations. The situation is shown in Figure 2. Also, it rarely happened that the latest iteration lasted a shorter or longer period than the other iterations. For this reason first and last iteration were removed from the results.

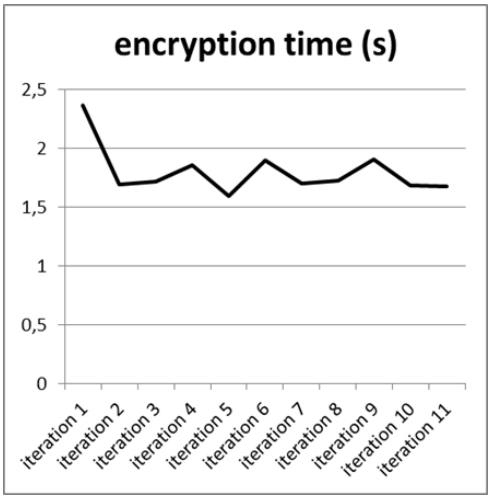

Fig. 2. The iterations AES encryption of 1 058 496 B file using phone LG P500

Problem No. 3: Theoretically System.currentTimeMillis() cannot be used for cryptographic tasks which have the encryption times shorter than 200 ms. Practically, the measurement accuracy may be 10 times to 100 times worse. That means that the measurements which use System.currentTimeMillis() should not be used for encryption tasks that are performed in less than 10 s [18].

Problem No. 4: System.currentTimeMillis() reflects the "wall clock time". If the system synchronizes time via NTP (Network Time Protocol) during the measuring, then this synchronization distorts the measurement result. The distortion affects all cryptographic tasks; it is not important that their execution takes less or more than 10 s [18].

Solutions of the problem No. 3 and No. 4: For this reason, all measurements were made using the API System.nanoTime. It can be used for shorter tasks than 10 s and "wall clock time" does not affect the benchmark results.

Problem No. 5: If the benchmark application uses huge GUI, for example ProgressDialog (for displaying the test status) or something similar, it can also distort the measurement accuracy.

Solution of the problem No. 5: Our cryptographic benchmark has only a minimal GUI, which consists of a single button. This button has only one function: It triggers the test. Any other visualization and interaction with the user was not built into the application. Benchmark sets all parameters of the test including the selection of files for testing, the number of iterations and all necessary paths. [18] The test results are automatically saved to the *.csv file after completion of the test.

V. Test Evaluation

In total, 6660 test files were encrypted. These devices were used for testing:

- LG P500 Optimus One (phone).
- Samsung Galaxy S4 i9505 (phone).
- Acer Iconia Tab A511 (tablet).

As mentioned earlier, both SD card and internal memory were used for saving encrypted files to the tested device.

Dependence of the encryption time and the file size is for all encryption methods approximately linear. Therefore, the regression lines were put over the dot plots.

A. Interpretation of the regression coefficient

The regression coefficient determines the speed of the encryption method. The value of the regression coefficient can be interpreted like this: increase in the file size by one megabyte causes increase in the encryption time value by regression coefficient (in seconds). The speed of methods ECC, AES and Triple DES on particular devices can be compared by using the values of the regression coefficients.

B. Interpretation of the regression constant

The values of the regression constant represent the time for preparation before the encryption itself.

The essential difference between the ECC method and the other two methods is that the former has a non-zero constant but constants of the other two methods can be considered to be a zero. In practice it means that the time of the ECC encryption is not insignificant even for files which have size close to a zero. These files have encryption time approximately equal to the value of regression constant. It is caused by the fact that the ECC method needs significant preparation time. Before the

encryption process, ECC method has to create a finite field, an elliptic curve, etc. Only after that, the ECC method can start with encryption. The measured results show that the ECC method is not suitable for small size files. On the other hand, the encryption time dependence on file size of the other two methods can be considered directly proportional. (The files which are insignificantly smaller have insignificantly shorter encryption time.)

C. Results of the encryption on Acer Iconia Tab A511 device

The AES method has the lowest regression coefficient and has insignificant regression constant. Therefore, the AES can be considered as the fastest method on this device. The comparison of the two other methods is not so clear. ECC method has a lower regression coefficient than TripleDES. Since the method ECC does not have insignificant time of preparation (internal persistent memory: 0.81 second, SD card: 0.78, see the values of the regression constants in Figure 3 and Figure 4), the ECCis not suitable for small size files. The Figures 3 and 4 show the intersections of the regression lines. This means that the encryption of the files up to 2.2 megabytes size is ECC method slower than the TripleDES method. For files which have larger size than 2.2 megabytes is the opposite situation. There the ECC method is faster than the TripleDES.

Both internal persistent memory and SD card have very similar encryption speed.

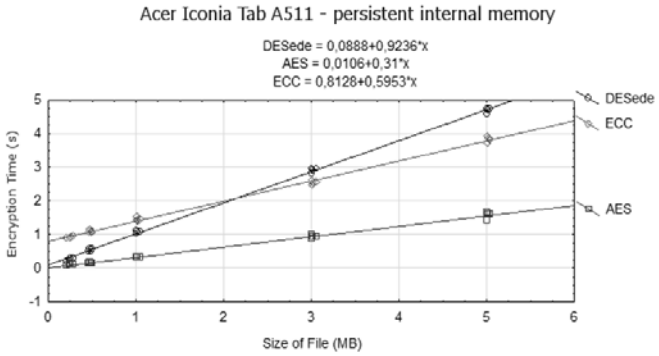

Fig. 3 The results of measurements on Acer Iconia Tab A511 device - persistent internal memory

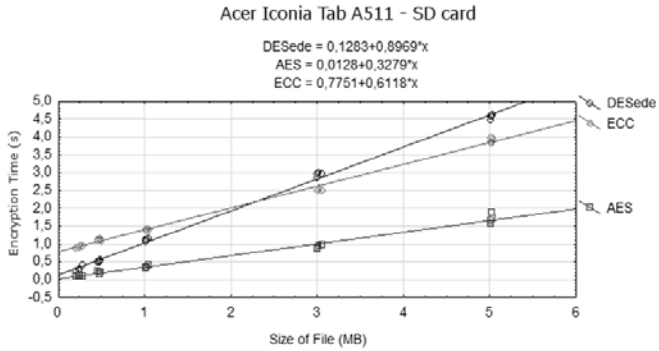

Fig. 4 The results of measurements on Acer Iconia Tab A511 device - SD card

D. Results of the encryption on LG P500 Optimus One device

The AES method has the lowest regression coefficient and has insignificant regression constant. Therefore, the AES can be considered as the fastest method on this device. The comparison of the two other methods is not so clear again. The ECC method has once again regression coefficient lower than the TripleDES. The ECC method takes a long preparation time - in this case the preparation time is 2.7 s.

Low computing power (CPU frequency, RAM frequency etc.) causes that the ECC method has a long preparation time. But effectiveness of ECC causes that the intersection of the regression lines is already at 0.8 megabytes. Therefore TripleDES is more suitable for files which have size up to 0.8 megabytes and ECC is more suitable for larger files.

The tests show that LG P500 Optimus One has much higher values of the regression coefficients compared to Acer Iconia Tab A511. Therefore the encryption speed of LG P500 Optimus One device is generally slower than the speed of Acer Iconia Tab A511.

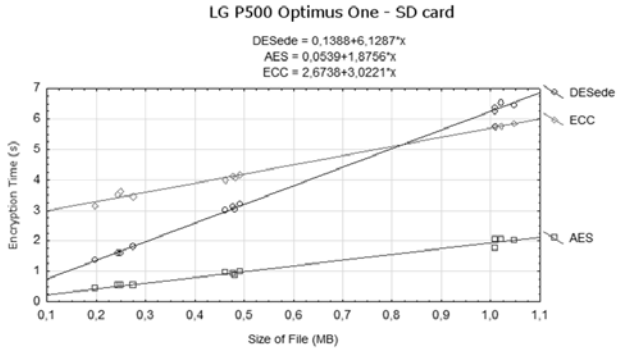

Fig. 5 The results of measurements on LG P500 Optimus One device - SD card

E. Results of the encryption on Samsung Galaxy S4 i9505 device

The test results which were made on Samsung i9505 Galaxy S4 were surprising. While AES and TripleDES methods significantly accelerated the encryption, the ECC method was decelerated. The ECC encryption on this device is slower in comparison with the Acer Iconia Tab A511. The values of regression coefficients 0.6 x (Acer Iconia Tab A511 - persistent internal memory) and 1.13 x (Samsung i9505 Galaxy S4) are not connected with the computing power of those devices. Because, according to benchmark performance [19], the device Samsung i9505 Galaxy S4 has more computing power than the Acer Iconia Tab A511.

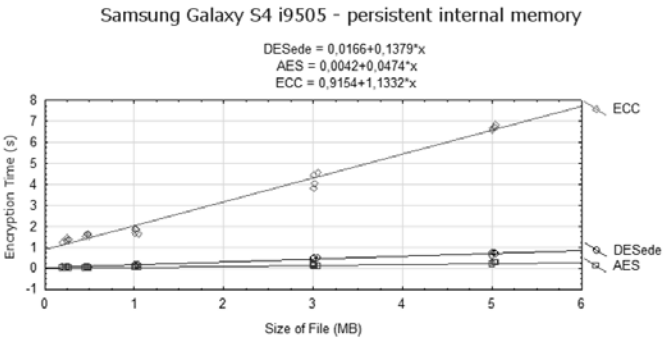

Fig. 6 The results of measurements on Samsung Galaxy S4 i9505 device - internal persistent memory

Hardware specs:

- Acer Iconia Tab A511 Chipset: Nvidia Tegra 3, CPU Quad-core 1.3 GHz Cortex-A9, GPU ULP GeForce.
- Samsung Galaxy S4 i9505, Chipset, Qualcomm APQ8064T Snapdragon 600, CPU Quad-core 1.9 GHz Krait 300,GPU Adreno 320.

There may be a lot of reasons for the bad ECC method results, i.e. TouchWiz or a different hardware architecture. It will be necessary to create special tests for resolving this problem. These tests will be developed as part of the research at the Faculty of Applied Informatics [20].

VI. Conclusion

The test results show that the AES method with a 256-bit key has the best encryption times for all files and on all devices. The AES method has good performance on Android mobile devices and at the same time AES is considered to be safe enough. For this reason, we recommend application programmers to use the AES method for the encryption in their applications. The length of a key should never be less than 256 bits. Due to the focus of mobile malware on the Android operating system as the primary target of attack, it is important for the application developers to use cryptography to the extent possible and protect the users and their data. Occasional time anomalies occurred during the tests. There were pairs of files of different types which showed a statistically significant difference in average encryption times on significance level of 5%. This raises a question of what causes this discrepancy. Detailed information about the time anomalies will be published. The cryptographic tests performed on a P-4 2.4 GHz machine at Washington University in St. Louis had similar AES and TripleDES results (the encryption time of AES was significantly better than the time of TripleDES) [21].

References

[1] Maly, V. Approach to teaching programming of applications for mobile devices. In *Recent Researches in Educational Technologies.* Corfu: WSEAS. 2011. pp 74-79. ISBN 978-1-61804-021-3.

[2] Samsung. Samsung Galaxy Tab® S 10.5" 16GB, Dazzling White. *Samsung.com* [online]. © 1995-2014 [cit. 2014-10-12]. Available: http://www.samsung.com/us/mobile/galaxy-tab/SM-T800NZWAXAR.

[3] Savu, L. Information Security on Elliptic Curves. In *Recent Researches in Communications and IT*. Corfu: WSEAS. 2011. pp. 69- 72. ISBN ISBN: 978-1-61804-018-3.

[4] Google. Google Play. *Play.google.com* [online]. ©2014 [cit. 2014-10-12]. Available: https://play.google.com/store

[5] Federal Bureau of Investigation. *(U//FOUO) DHS-FBI Bulletin: Threats to Mobile Devices Using the Android Operating System.* [online]. 2013. Washington: Department of Homeland Security, Federal Bureau of Investigation. Available: https://publicintelligence.net/dhs-fbi-android-threats/

[6] Hankerson, D., Menezes, A. and S. Vanstone. *Guide to elliptic curve cryptography*. Springer, 2004, 312 p. ISBN 038795273X.

[7] Gwalani, K. A., Elkeelany, O. Design and Evaluation of Hardware Accelerator for Elliptic Curve Cryptography Point Multiplication. In *Recent Advances in Applied Mathematics and Computational and Information Sciences - Volume II*. Houston: WSEAS. 2009. pp. 431-436. ISSN 1790-5117.

[8] Announcing the Advanced Encryption Standard (AES). *Federal Information Processing Standards Publication.* ©2001. 51p. [online]. ©2014 [cit. 2014-01-20]. Available: http://csrc.nist.gov/publications/ fips/fips197/fips-197.pdf.

[9] EAST-TEC. Use the U.S. Government approved algorithm for storing classified information. East-tec.com. [online]. © 1997-2014 [cit 2014-10_12]. Available. http://www.east-tec.com/kb/what-is-the-aes-standard/

[10] Data Encryption Standard (DES). *U.S. Department of Commerce/National Institute of Standards and Technology*. [online]. © 1999. [cit. 2014-01-20]. Available: http://csrc.nist.gov/publications/ fips/fips46-3/fips46-3.pdf.

[11] Ouleha, M. *Šifrování dat na mobilních zařízeních Android.* Zlín: Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky. Diplomová práce, 177 p. Available: http://dspace.k.utb.cz/handle/ 10563/24705.

[12] Hi-Q MP3 Voice Recorder. *Google Play* [online]. ©2014 [cit. 2014-01-20]. Available: https://play.google.com/store/apps/details?id=yuku.mp3 recorder.lite&hl=en.

[13] Simple Notepad. *Google Play* [online]. ©2014 [cit. 2014-01-20]. Available: https://play.google.com/store/apps/details?id=org.mightyfrog. android.simplenotepad&hl=en

[14] Java™ Platform Standard Ed. 6. Class Cipher. *Docs.oracle.com* [online]. © 1993-2011 [cit. 2013-01-25]. Available: http://docs.oracle.com/javase/6/docs/api/javax/crypto/Cipher.html.

[15] LG Optimus One P500. *GSMARENA* [online]. ©2000-2014 [cit. 2013-01-30]. Available: www.gsmarena.com/lg_optimus_one_p500-3516.php

[16] Java™ Platform Standard Ed. 7. Class OutOfMemoryError. *Docs.oracle.com* [online]. © 1993- 2014 [cit. 2013-01-30]. Available: http://docs.oracle.com/javase/7/docs/api/java/lang/OutOfMemoryError. html.

[17] Managing Your App's Memory. *Android Developers* [online]. [cit. 2013-01-30]. Available: http://developer.android.com/training/articles/ memory.html.10563/24705.

[18] Boyer, B. *Robust Java benchmarking, Part 1: Issues. Understand the pitfalls of benchmarking Java code.* 2008, 18p. Available: http://www.ibm.com/developerworks/library/j-benchmark1/j-benchmark1-pdf.pdf.

[19] Qualcomm Snapdragon S4 Pro (APQ8064) vs Nvidia Tegra 3 (T30). *CPUBoss* [online]. [cit. 2013-01-30]. Available: http://cpuboss.com/cpus/Qualcomm-Snapdragon-S4-Pro-(APQ8064)-vs-Nvidia-Tegra-3-(T30).

[20] Tomas Bata University in Zlín. The Faculty of Applied Informatics. Available:http://www.utb.cz/fai-en.

[21] Performance Analysis of Data Encryption Algorithms. *Washington University in St. Louis* [online]. ©2014 [cit. 2014-03-09]. Available: http://www.cs.wustl.edu/~jain/cse567-06/ftp/encryption_perf