The Sources of Randomness in Mobile Devices

Jan Krhovják, Petr Švenda, Václav Matyáš
Masaryk University, Brno, Czech Republic
{xkrhovj, xsvenda, matyas}@fi.muni.cz

Abstract

The goal of this paper is to examine randomness sources available in current mobile phones or other mobile devices. We identify potential sources of randomness and perform an analysis focused on the camera and the microphone input noise as promising sources of randomness. We also perform statistical tests and analyse quality of these sources of randomness including estimation of entropy in the generated data.

Keywords: Entropy estimation, mobile device, smartphone, source of randomness.

1 Introduction

Our paper deals with issues related to the generation of truly random data (i.e., bits, numbers, and sequences) in mobile computing environments. We describe the expected application requirements in terms of amount and speed of random data generation in such environments, and then we focus on available sources of randomness in mobile devices. The main goal of this work is identification of such sources, evaluation of their acquisition speed, statistical testing of their quality, and estimation of the amount of available entropy in a given time period. Some issues described in this paper were discussed at the conceptual level in [5].

Our work covers the possibilities of random data generation from both external and internal environmental characteristics. The examined external sources are audio and video streams that can be captured by a mobile device. With respect to the internal sources, we investigated the information accessible through the application programming interfaces (API) of mobile devices like the actual battery charge level and other accessible system statistics.

Identified sources of randomness are tested on the Nokia N73 with the Symbian OS and E-Ten X500 and M700 with Windows the Mobile OS.

2 Basics of random number generation

Overwhelming majority of commonly used cryptosystems is designed in accordance with the Kerckhoffs' principle – i.e., the security of cryptosystem must be based on secrecy of the cryptographic keys and not on the a cryptosystem itself. This allows anybody to perform an independent analysis of particular cryptosystems (ideally also alongside with the verification of their source codes) and on the contrary, security can essentially be reduced reduced to the secrecy of private/secret keys. The quality and unpredictability of random data (i.e., bits, numbers, and sequences) that are the basis for cryptographic keys are therefore critical. Random data are in addition to generation of cryptographic keys also used for other cryptographic operations – e.g., as initialization vectors, various pad values

(padding), random challenges (nonces) in cryptographic protocols, in the process of digital signing (per-message secrets), and for masking data to prevent dangerous side-channel attacks.

In general, two kinds of generators can be distinguished – the true random number generator and the pseudorandom number generator. The former is typically based on nondeterministic physical phenomena (e.g., radioactive decay or thermal noise), while the latter is only a deterministic algorithm where all randomness of the output is fully dependent on the randomness of the input (often called seed). Getting truly random data in the deterministic environment of computer systems is extremely hard and slow (i.e., only a small amount of good quality random data can be generated in a reasonable time), therefore we often restrict ourselves to the use of deterministically generated pseudorandom data. Generating pseudorandom data is typically (in most environments) faster and truly random data are used in this process only as an initial input. Since the whole generating process is deterministic, the randomness of the output is fully dependent on the randomness of the input. Many classes of pseudorandom number generators (designed, e.g., for simulation purposes or randomized algorithms) exist, but the goal of a pseudorandom number generator in cryptography is the production of pseudorandom data that are computationally indistinguishable from truly random data. The goal of cryptanalysis is to prove the converse.

Randomness is a probabilistic property, therefore verification of the statistical quality of (pseudo)random data by detecting deviations from true randomness (known a priori) is based on statistical testing. These tests may be useful as the first step in determining whether or not a generator is suitable for a particular cryptographic application. However, no set of statistical tests can absolutely certainly point out a generator as appropriate for usage in a particular application, i.e., statistical testing cannot serve as a substitute for cryptanalysis. In addition, the results of statistical testing must be interpreted with some care and caution to avoid wrong conclusions about a specific generator. Most commonly used statistical test suites are DIEHARD [4] and NIST [1], which test whether the presented sequence statistically appears in the same way as the truly random sequence would.

2.1 Specifics of the mobile devices

Mobile devices are different from general purpose computers and this also influences the process of generating of (pseudo)random data. The possibility to change environment of mobile devices is definitely a great advantage given by the mobility nature of the device. The existence of several embedded input devices as microphone, radio receiver, video camera, or touchable display is another advantage. On the contrary, the mobility and the small physical size of device brings also a higher possibility of a theft or (temporary) lost with potential compromise of generator. The important assumption of high-quality secure generator design is thus the impossibility of deducing the inner state of the generator and fast recovering of its entropy level (after time-limited compromise). Other disadvantages can be low performance (CPU frequency is in the best smartphones roughly 200MHz) and restricted random access memory size (in the order of tens of MB).

Application requirements are both static (e.g., cryptographic keys) and dynamic (e.g., initialization vectors). The latter case is restricted (and bounded) by the type used transmission technology (EDGE, Bluetooth, WiFi, WiMAX, etc.). However, exact requirements here are dependent also on the application in use that can be categorised as an interactive (e.g., transfer of voice or video), semi-interactive (e.g., web-based services), and non-interactive (e.g., one-time file transfer or sending e-mail). The data can be thus encrypted and transferred (according to these categories) immediately after creating/filling outgoing packet (to prevent unwanted delays), or after reaching other pre-specified sizes. A new initialization vector (and often also padding) is required for each encryption. Splitting data to several independent pieces thus impllies higher requirements on initialization vectors (and padding). The consequence is higher requirements on (pseudo)random data.

Well-designed and robust generator must always have a sufficient amount of entropy – shortly after turning the device on, after letting the device out of sight, and after an intensive generation of random data. This non-trivial task may require employment of the energetically costly sources of randomness (e.g., video camera) and/or the user contribution. Utilization of these sources may be required to assure higher security of generated data – e.g., for mobile banking purposes.

Important characteristics of all sources are: availability in different environments, time variability with noise presence and unpredictability and uninfluenceability for the attacker. More detailed discussion of possible sources of randomness and attacker models can be found in [5].

3 Analysis of selected sources of randomness

This part of the paper deals with issues related to the practical experiments performed on the smartphone Nokia N73. The goal of these experiments was to assess the quality of selected randomness sources in one particular mobile device and to estimate the amount of randomness (entropy) in these sources. Due to the API restrictions, we were forced to drop sources like the battery level, signal strength or GPS position as measurements over these sources do not provide an output with a sufficient precision (e.g., battery and signal values are available in the form of an integer between 0 and 10) or frequency (e.g., external GPS provides only one measurement per second with fluctuations typically only in two least significant bits).

3.1 Theoretical entropy estimation

The basic measure for randomness is in information theory often called uncertainty or entropy and is typically defined [6] as:

$$H_1(X) = -\sum_{x \in X} P_X(x) \log P_X(x)$$

where the sample x is drawn from random distribution X with probability $P_X(x)$. The logarithm base typically correspondents to the unit of measured information – in information theory base 2 is often used and that implies that the unit will be bits. This entropy measure is often referred as Shannon entropy or alternatively information entropy.

Unfortunately, Shannon entropy may be inappropriate for our purposes, since it is in fact only average case entropy. We cannot make any assumptions about distributions formed by our sources of randomness. The problem is that the attacker can simply force the source of randomness to produce the most probable values that contain minimum entropy. To cope with this situation the min-entropy measuring the worst case entropy is often used (especially in the theory of randomness extractors). It is defined as:

$$H_\infty(X) = min_{x \in X}(-\log P_X(x)) = -\log(max_{x \in X} P_X(x))$$

where the sample x is drawn from random distribution X with probability $P_X(x)$. It can be easily seen that min-entropy is always less or equal then Shannon entropy (the tight example is for uniform distribution). These two entropy measures are special cases of generalised so-called Rényi entropy [2].

3.2 Microphone input

An embedded or hands-free microphone is used as a voice input device in mobile devices. Almost all commonly used microphones are typically based on an oscillating membrane and some mechanisms that transform the oscillation to the voltage representing particular signal elements. Supported sampling frequency, modulation method, and number of bits used for representing the value of one sample

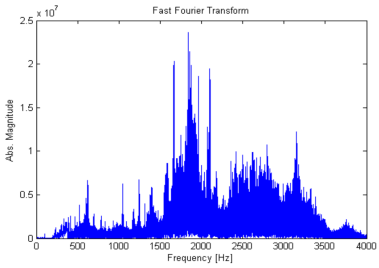

Figure 1: Frequency spectrum of the recorded music sample (embedded microphone).

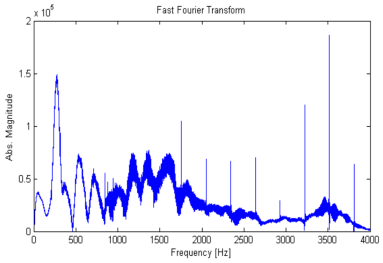

Figure 2: Frequency spectrum of the recorded noise sample (embedded microphone).

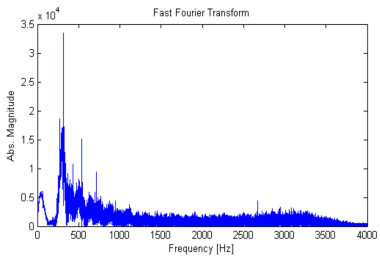

Figure 3: Frequency spectrum of the recorded noise sample (hands-free microphone).

are the most important parameters of such devices. The Nokia N73 smartphone uses 16-bit pulse coded modulation (a signed PCM) at the frequency 8000 Hz for sampling a sound wave – the data throughput is thus 16000 B/s. We restrict ourselves only to the small number of 204800 samples that corresponds to 25.6 seconds of sound due to memory restrictions of the inspected device.

Note that each microphone may have slightly different characteristic (e.g., due to different solidity of membrane or other manufacturing differences). Our goal is to estimate the amount of entropy in the input sound signal captured by the microphone. We focus mainly on measuring min-entropy of the noise originated in the microphone.

We used the fast Fourier transformation (FFT) algorithm to compute a discrete Fourier transform (DFT) for analyzing the basic frequency components present in the noise. We performed this analysis of the embedded microphone on the sound sample of both recorded music (Figure 1) and noise (Figure 2). Moreover, we analysed also the hands-free microphone on the sound sample of noise (Figure 3). We are interested only in the spectrum of frequencies between 0 and 4000 Hz due to the Nyquist theorem (as we are sampling at 8000 Hz). Ideal noise is expected to have all frequencies uniformly present. The results shows that there are significant differences in the observed spectrums of the noise. The hands-free microphone appears to be more sensitive than the embedded one.

Finally, we analysed histograms of all these recordings. They have approximately normal distribution and the actual minimum/maximum values are –32764/32767, –14220/8856, and –347/574, respectively. However, especially in the case of noise, these numbers are strongly influenced by the sharp peak on the beginning of each recording trace – caused probably by the device turn on. More accurate limit values –12/13 and –9/8 are obtained when this peak is removed. Hence, we can make a very preliminary estimation that at most $log_2 8 = 3$ bits of entropy can be presented in each sampled value. Of course, this is only the upper bound of the maximum possible entropy.

Focusing only on the more sensitive hands-free microphone, the estimations are 2.9 bits of entropy according to the Shannon formula and 0.5 bits of entropy according to the min-entropy formula. These estimations (and especially probabilities used in those formulas) were also calculated from histograms with the assumption of full independency within the samples. However, correlation tests (described below in camera section) found correlation in the recorded samples of noise. This correlation decreases as we took only every second/third value from our samples. Sequence created from every fourth value was without statistically significant correlations. We therefore recommended to lower estimated entropy at least 5× with respect to amount calculated for each sample value.

3.3 Camera input

Digital optical input devices (e.g., cameras, microscopes, or scanners) can be based on several different silicon sensors (e.g., CCD, CMOS, EMCCD, and ICCD) [7]. All of them use an array of semiconductor

photo-sensors to transfer an accumulated electric charge to a voltage. Digital cameras based on CCD (e.g., Sony-Ericsson S700i) or CMOS (e.g., Nokia N* series) sensors (for simplicity will be denoted as *optical sensors*) are commonly used in current computer systems and mobile phones. As we will mention in next paragraphs – these optical sensors are influenced by thermal noise, they have problems with vignetting, blooming, sensitivity to some colours, etc. Some of these problems are solved by the manufactures by purely software means (that are often kept secret). This makes the entropy estimation harder.

3.3.1 Camera view finding noise

A significant part of current mobile devices (cell phones, smart phones and PDAs) is equipped with a built-in camera that can be used for entropy gathering. For camera input we suggest that entropy should be extracted from an optical sensor noise during "view finding" rather than from a high-resolution picture of the surrounding environment. The data output from view finding is more suitable source than a high-resolution snapshot output for two main reasons: 1) view finding is not post-processed by software noise reduction and compression, 2) data acquisition is much faster – commonly between 10–15 frames per second – and has more suitable size for the additional manipulation: a single frame with 180×240 pixels can be fully stored and processed in the RAM memory (∼130kB), which is unlikely for a high-resolution picture.

Lower-quality optical sensors (often used in mobile devices) have generally higher noise presence than sensors in high-end cameras. The noise is typically unwanted for almost all applications – with the exception of the random number generation process. The optical sensor white noise should be always present, but its actual level may depend on physical conditions – namely the temperature. We performed practical experiments with the Nokia N73 camera within temperatures 5 ℃ and 45 ℃ and concluded that an inside decrease of the noise towards lower temperature can be detected. However, this noise is still significantly present to provide enough entropy.

The input source is available even when the camera cover is closed or lens are covered with a finger. This is both convenient and useful – it serves as an important defence against an active attacker that illuminates the sensor. An overexposure of the sensor can be caused, for example, by an intense light, like the halogen lamp (see Figure 4). All colour components within the exposed area are boosted up to the maximum value (255) and thus all possible entropy obtainable from (not only white) noise is effectively removed. A similar effect can be also caused by direct sunlight. To successfully mount such attack, attacker must be able to overexpose almost the whole area of the camera chip. Degradation during random data generation is prevented if the lens are equipped with a closable cover or are shielded by the finger. This assumption is reasonable when the device is controlled by the user and an attacker can only manipulate surrounding environment. On all accounts, the random data gathering application should test the input stream for overexposed values and lower the estimated amount of gathered entropy in real-time.

3.3.2 Camera input entropy estimation

In this section, we would like to estimate the amount of entropy extractable from a camera input. We proceed as follows:

1. We developed a custom application for storing large amount of frames produced by a mobile device camera into a removable memory card. Systematic defects of the input due to post-processing were examined using visualization and statistical tools.
2. Noise dependency on the surrounding temperature was experimentally measured and evaluated.

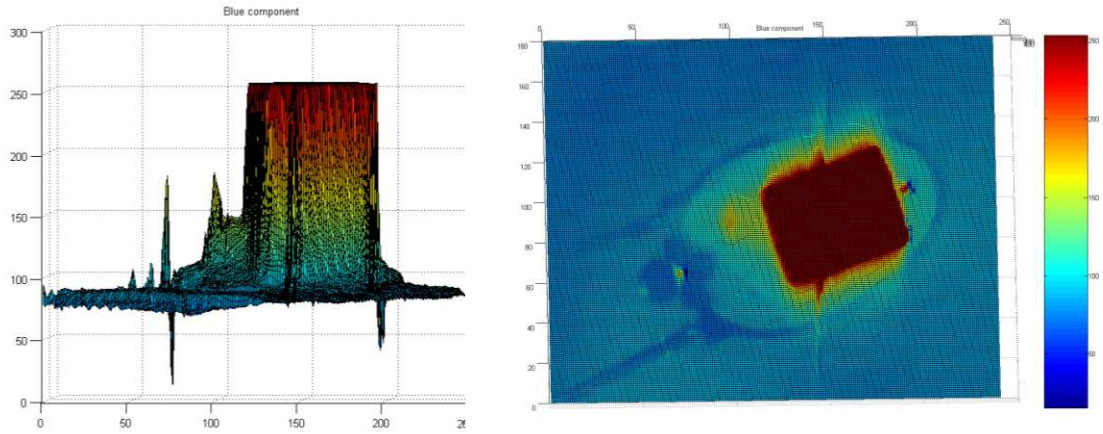

Figure 4: A visualized overexposure caused by the halogen lamp (blue colour component).

3. The correlation between neighbour pixels, pixels in the same row and column was computed, as well as autocorrelation and fast Fourier transformation of values from single pixel in time (subsequent frames).

4. The distribution of values for separate colour components was computed for the temperature with the least noise within device operation value (5 ℃) and entropy was estimated based on Shannon entropy and min-entropy formulas.

5. A simple technique for entropy extraction (with least possible post-processing) was implemented and we tested the resulting binary stream tested with the suite of NIST randomness tests to evaluate its statistical properties.

6. We performed, practical tests of maximal data throughput of SHA-1 hash function on mobile devices with Symbian OS and JavaME. Preliminary entropy extraction function using SHA-1 was implemented.

The systematic defects introduced into camera frames due to post-processing and optical sensor technology (like row-dependent readouts) are clearly visible from Figure 5 when the average camera input with a closed lense cover is visualized. There are hot pixels around borders, significant rips in the rows, centered circle rips and significantly different intensity towards centre of the frame. Especially, blue colour component shows visible row-dependent rips (caused by the readout technology) and red colour component has significantly increased value towards chip borders. This effect is probably caused by camera post-processing (out of our control) to balance light drop due to different lens mass towards borders. Such effect may lead to a suspicion that some pixels are systematically correlated. However, when values for each single pixel are normalized by subtracting mean of the pixel in a longer time period, most systematic effects vanish. The colour histograms of normalized input taken over 2000 subsequent frames are depicted on Figure 6 (histograms are centred to 0). All colours exhibits Gauss-like distribution, blue colour providing more entropy (in Shannon sense) than other colour components. The presented frames were taken in a temperature around 5 ℃.

A well documented property of the optical sensors is dependency of noise on the temperature – the noise component should be reduced by lower temperatures (e.g., under-cooling of chip by liquid nitrogen). We tested the noise presence with a lens plastic cover in temperatures of 5 ℃ and 45 ℃ (more precise measurement of optical sensor is not possible without depackaging). The detected differences between temperatures were detectable, but negligible[1] and we can expect slightly decreased (but still comparable) amount of entropy for the lower temperatures. We used noise obtained from measurements in 5 ℃ for entropy estimation. In the common camera devices the raw data from the optical

[1]Difference in values probability only in order of 1/1000.

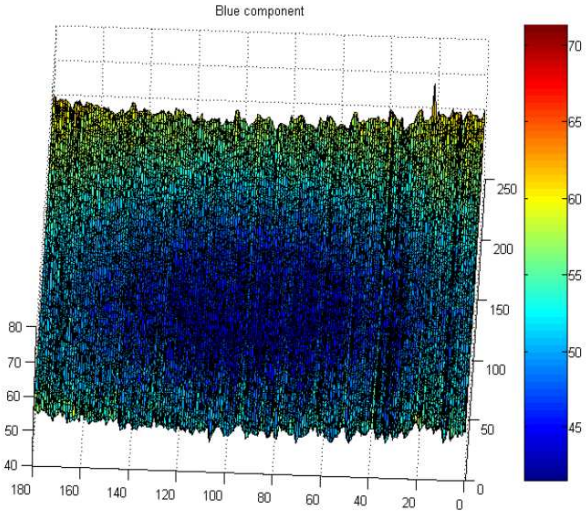

Figure 5: The average value of the blue colour component over the whole camera's frame with closed cover.

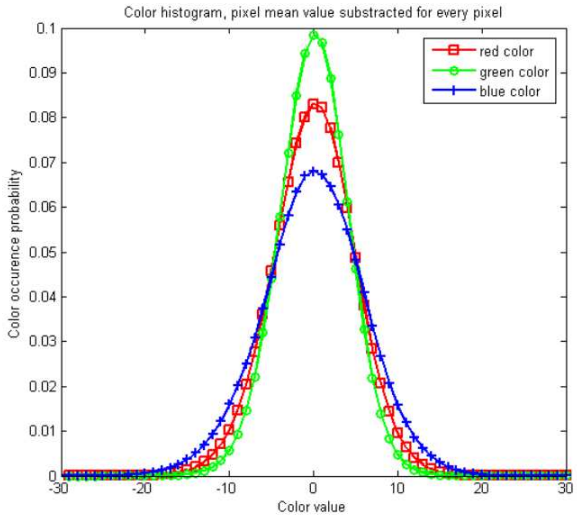

Figure 6: The intensity histogram for each colour component after remove of the pixel post-processing effects by normalization.

sensor are not directly accessible to the user, but they first go through a few correction steps, which are usually incorporated into the firmware of the optical sensor. Such corrections take care of some of different systematic effects, which are partially inherent to the optical sensor working principle and partially occur by fabrication tolerances of the manufacturing process.

For an overall entropy estimation, we have to know the number of independent (uncorrelated) pixels in one frame and whether values of single pixel are independent between two or more subsequent frames (or which frames has to be discarded to obtain uncorrelated values (e.g., pixel value from every 3rd frame) and finally estimate the expected entropy provided by single independent pixel.

A cross-correlation function (Matlab *corrcoef*) was used to verify whether neighbouring pixels and pixels in the same row and column are independent. No statistically significant correlation was found on significance level set to 0.01[2]. We generated streams of a binary data from red, green and blue colour component and tested those streams with the NIST battery for a further verification of results obtained from cross-correlation. See section 3.3.3 for details.

The camera view finding mode on Nokia N73 provides us with 12 frames per second and thus 12 different values per single pixel (10–15 frames for other cameras). The key question for entropy estimation is the independency of consequent values of that pixel. The auto-correlation tests were performed with the vector containing the values taken in time from a single pixel. Statistically significant deviations from the characteristics of white noise (where significant correlation value is present only for $lag = 0$ and can be omitted otherwise) were not detected. This implies that the consequent frames should be independent. All view finding pixels (180×240) were separately tested within a sequence of 2100 frames. Fast Fourier transformation was applied to the same vector (values of a single pixel in time) to detect dominant frequencies. FFT provided almost uniform output (no dominant frequency) for all tested pixels and thus confirmed independence of values between frames from an auto-correlation test. Note that as view finding provides 12 frames per second, only frequencies between 0 to 6 Hz can be tested by FFT due to the Nyquest theorem.

The final step is to compute the amount of entropy carried by the noise in a single (independent) pixel. As can be seen from Figure 6, histograms of all colour components follow the common Gauss

[2]This result means that the number of tested vectors with a correlation coefficient lower than 0.01 (strong correlation) is not significantly higher than the number of correlated ones between truly random vectors. It means that only around 10 vectors from 1000 are expected to be correlated for this significancy level.

	Single pixel (Shannon)	Single pixel (Minimal)
Red colour (5 ℃)	3.9203	3.1979
Green colour (5 ℃)	4.0373	3.3277
Blue colour (5 ℃)	4.7608	3.9276

Table 1: An entropy estimation according to Shannon and min-entropy formulas for a single independent pixel. Note, that extrapolation to whole input (180×240 pixels/frame, 12 frames/second) is valid only if the pixels and frames are independent (see 3.3.2 for discussion and performed tests).

distribution without significant deviations. Overall entropy (see Table 1 for EntropyPerPixel) of camera view finding source can be computed using a simple formula [bits/s]:

$$E = IndependentPixelsPerFrame * IndependentFramesPerSecond * EntropyPerPixel$$

3.3.3 Statistical testing with the NIST battery

As described in the previous section, we also wanted to verify the statistical properties of binary stream extractable from view finding input. We extensively tested many sequences extracted from the input by various different techniques (incorporating only particular colours, pixels, and/or utilising simple digital post-processing). Our goal was to find the easiest way to extract binary sequences that will pass common statistical tests. This should help us to assess how much entropy can be present in input data that we will use for seeding a pseudorandom number generator based on classical cryptographic functions. Notice that our techniques are far away from theory of classical secure entropy extractors (a stimulating discussion can be found in [3]) that are constructed by a considerably different way.

Here we should briefly refresh how the NIST test suite works. The input sequence is first divided to some number of subsequences and to these subsequences are applied several statistical tests – each yielding so called *p-value*. There are two ways of interpreting the results of tests. The first method evaluates the proportion of successfully passed subsequences (successful pass means that the *p-value* is less then selected significance level) and the second evaluates uniformity of *p-values* (using chi-square goodness-of-fit test).

Tested extraction methods:

1. Raw values only – we begin with testing the raw input for concrete nine pixels and its basic colours (in both cases separately) and all used NIST tests always failed. The reason is that there is an obvious non-uniformity of values in particular colours and thus also in the whole pixel (that consists of three basic colours saved one after another). The statistical distribution of its values is depicted on the Figure 6.

2. Least significant colour bit – we then extract only the least significant bit from each pixel and colour (one bit per frame is extracted from the single pixel). All binary strings (for each pixel and its colour) constructed this way also failed the tests. All tested sequences constructed from least significant bits (LSB) failed both the proportion of successfully passed subsequences and the uniformity of *p-values*. However, for each pixel/colour there was a part (40–60%) of subsequences that passed the runs, frequency and cumulative-sums test. This proportion was still too small to pass the whole test, but these tested sequences have slightly better statistical properties.

3. Colour value combination using XOR – we XORed all three colours together but it was no surprise that NIST tests also failed for all nine pixels. We also tried to construct the sequence

from bits obtained by XORing all eight bits of each value of colour. The results were similar as in the case of using LSB. We conclude that these elementary techniques can not suppress all statistical defects introduced in sequences constructed from one pixel and its colours and utilising of more pixels is thus required.

4. Flip-flop bit extraction – simple but more robust technique that extracts one bit per one pixel colour component. Extracted bit is 0 or 1 when actual colour value is odd or even. Concrete mapping between the colour and bit value is reversed after each processed pixel.

Let us describe the flip-flop extraction technique in more details. Note that the purpose of te flip-flop extraction is not a new extraction technique for real usage, but to design simplest possible technique used only for our statistical testing. This technique then extracts some entropy from the input, but does not propagate it through the sequence (confusion in the cryptographic sense) like, e.g., SHA-1 does. Flip-flop is used only during entropy estimation, not in an implementation of a real generator (where, e.g., SHA-1 may be used). For a given frame, all pixels are processed row by row:

1. For every even pixel do: if pixel value mod 2 = 0, then set output bit b to 0 otherwise to 1.
2. For every odd pixel do: if pixel value mod 2 = 0, then set output bit b to 1 otherwise to 0.
3. Bit b is appended at the end of the bit stream.

Set of the possible values from the range 0–255 is divided into two groups. The first group contains only even values and the second group contains only odd values. Both groups should have almost equal sum of probabilities. Values from the first group will result in bit value 0, second group in value 1. Unfortunately, separation into two groups with the same probability is not possible. As groups will not have exactly the same probability, fixed bit assignment rule may result in a significant difference between number of ones and zeroes. We chose to invert this bit value for half of the pixels thus balancing more probable value 0 for one pixel by a higher probability of 1 for the next one and vice versa.

The independent binary streams were constructed for each colour component using described extraction technique with all pixels within the frame (2100 frames were used, gathered in a burst of 7 consequent frames followed by approximately a 5 second delay needed to save data to removable card). All streams were tested with the NIST battery (100 sequences per 1 Mb) at the significance level 0.01. One sequence failed for the red colour, all passed for the green colour and two sequences failed for the blue colour (one template in non-overlapping template tests). We are aware of the possible impact of relatively short length of sequences, but we were restricted by the camera memory, acquisition speed (especially time needed to store the captured frame on a removable card) and 1 bit per pixel extraction technique.

4 Results for PDA phones E-Ten X500 and M700

In this section we summarize our results from additional experiments with PDA phones Glofish E-Ten X500 and Glofish E-Ten M700.

4.1 E-Ten specifics and capturing of picture

E-Ten PDA phones X500/M700 are based on the Windows Mobile 5 OS (WM5) that can be easily upgraded also to the Windows Mobile 6 OS (WM6). These operating systems support a special native Camera API for a direct camera control. Unfortunately, this low-level API is in turn not supported

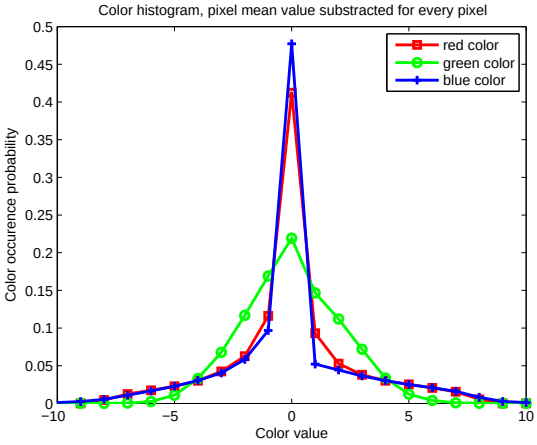

Figure 7: E-Ten X500 – Y-axis values are in the magnitude of 10^7.

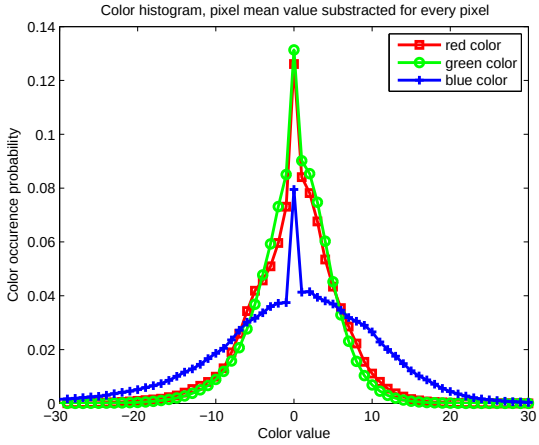

Figure 8: E-Ten M700 – Y-axis values are in the magnitude of 10^6.

by the tested E-Ten devices (regardless of the OS, tested for both WM5 and WM6) and so we used a special video driver capable of capturing and sending the actual screen image over the USB port to a PC. This approach has its advantage – the data can be retrieved and stored faster than for the N73 phone and the total amount of the captured data is not limited by the phone flash memory. However, there are also a drawbacks – transferred pictures have only a limited number of possible colour values (red and blue colour components are limited to 5 bits per pixel; green colour component is limited to 6 bits per pixel – while the original resolution was 8 bits per pixel) and can only be retrieved at the rate of one picture per second.

4.2 Camera input

The first significant difference to N73 is that the camera of E-Ten devices is automatically turned off when the lens are covered. This would imply that a direct capture of noise on a perfectly black background is not possible (see the following section for clarification). E-Ten X500 and E-Ten M700 also have different camera chips with visually different noise patterns. This was confirmed by the statistical analysis of the intensity histograms for each colour component after removing the pixel post-processing effects – see figures 7 and 8 with histograms of distances from the mean value. The significant peak at zero axis (mean value) for all colours means that actual colour value of the pixel often tends to be equal to the mean value. This can be caused by a limited precision of the captured images.

4.2.1 Camera input entropy estimation

The noise produced by a camera chip is again dependent on the temperature, similarly as for the N73 device. At 45 ℃ the noise from the chip is so significant that prevents a software switch off even when the camera lens are completely covered. Unfortunately, the threshold temperature for this behavior is around 35 ℃ and thus not reachable during the device standard operating temperatures. However, it allows us to obtain the noise from a black-only input, similarly to the scenario with the closed cover in the case of N73.

Distributions of the noise (presented above) are taken from the images captured in an 8 ℃ environment as this is the scenario with lowest noise presence within common usage operational conditions.

	Single pixel (Shannon)	Single pixel (Minimal)
X500 Red colour (8 °C)	3.0851	1.2786
X500 Green colour (8 °C)	3.1220	2.1894
X500 Blue colour (8 °C)	2.9927	1.0675
M700 Red colour (8 °C)	4.4626	2.9875
M700 Green colour (8 °C)	4.2577	2.9284
M700 Blue colour (8 °C)	5.3759	3.6527

Table 2: Entropy estimation according to Shannon and min-entropy formulas.

Both cameras of our E-Ten devices probably utilize an automatic ISO sensitivity correction – with the same temperature, more noise can be obtained if the light conditions are worsened (decrease of ambient light or an object put close to the lens). As the ISO sensitivity is automatically increased by the camera controller, the noise generated by the chip is amplified and more noise is present in captured images. This hypothesis based on visual observations was tested experimentally. The camera chip was heated up to 50 °C at which the camera input is not turned off by the postprocessing and series of the images were taken. The temperature was then lowered to 40 °C with the lens still covered and next series of images were taken. At the hardware level we see (as expected) that the noise level decreases with a decreasing temperature. But the resulting noise present in the captured images exhibits an opposite trend – there is more noise for the lower value. This seemingly opposite result is caused by the automatic ISO correction. As the real noise level produced by the chip decreases with the decreasing temperature, camera controller (out of our control) detects a smaller range of colour values at the actual ISO level and automatically increases the level. Increased ISO level results in an amplification of the noise generated by the chip and thus more noise is propagated into the captured picture.

Estimates of entropy present in samples from cameras of E-Ten X500 and M700 are summarized in the Table 2.

4.3 Microphone input

Embedded microphones in the E-Ten devices are more sensitive and thus also capable to record significantly more noise than the microphones used in the Nokia N73. Basic frequency components present in the noise are depicted bellow.

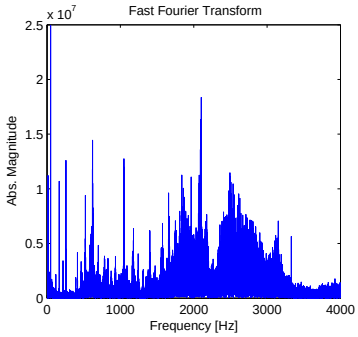

Figure 9: M700 – recorded music (same as for N73).

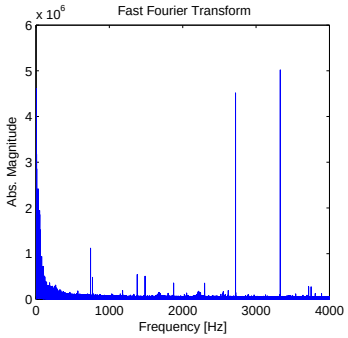

Figure 10: M700 – recorded noise sample.

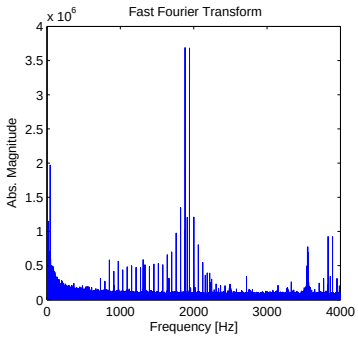

Figure 11: X500 – recorded noise sample.

The histograms of all these recordings have approximately normal distribution and the actual minimum/maximum values are –32258/32767, –1958/1633, and –806/716, respectively. There are no peaks

at the beginning of each recording trace. Hence, we can make a very preliminary estimation that at most 11 bits of entropy (given by encoding) can be extracted from each sampled value. Of course, this is (again) only the upper bound of the maximum possible entropy. The precise estimations are 10.124 and 9.365 bits of an entropy according to Shannon formula and 0.016 and 0.023 bits of entropy according to min-entropy formula for M700 and X500, respectively.

5 Conclusions and future work

We have seen that mobile devices provide us with several randomness sources that can be utilized for generating truly random numbers. Namely the microphone and camera inputs that are available in (almost) each mobile phone have a promising potential and should allow for generating data with a sufficiently large amount of entropy that can be used for cryptographic purposes. A subject of our next research will be a better examination of these sources with more accurate entropy estimation, particularly the mask of independent elements from a given source (pixels for a camera, sampled values from a microphone) to estimate an extractable random bits per second.

References

[1] Federal Information Processing Standards Special Publication 800-22. A statistical test suite for random and pseudorandom number generators for cryptographic applications, available at: http://csrc.nist.gov/publications/nistpubs/800-22/sp-800-22-051501.pdf. 2001.

[2] Rényi A. On measures of information and entropy. In *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability*, pages 547–561, 1960.

[3] Barak B., Shaltiel R., and Tromer E. True random number generators secure in a changing environment, available at: http://theory.csail.mit.edu/∼tromer/papers/rng.pdf.

[4] Marsaglia G. Diehard statistical tests, available at: http://stat.fsu.edu/pub/diehard/. 1995.

[5] Krhovják J., Švenda P., Matyáš V., and Smolík L. The sources of randomness in smartphones with symbian os. In *Proceedings of Security and Protection of Information 2007, Brno*, 2007.

[6] C. E. Shannon. A mathematical theory of communication. In *The Bell System Technical Journal*, 1948.

[7] Andor Technology. Digital camera fundamentals, available at: http://www.andor.com/pdfs/digital camera fundamentals.pdf.