Image Encryption in Medicine

Gabor Szivos – 1227443
Georg Bernold – 1325845

Seminar aus Medizinischer Informatik

Contents

ABSTRACT

1 Introduction

introduction to the general topic Privacy is one of the key issues of the information age. Especially in the post-Snowden era this fact has become clear to the most individuals around the globe. Numerous regulations and laws [1, 2, 3, 4, 5] require it across all kinds of use cases and industries. A well known way to achieve privacy is through encryption, which means encoding an information in such a way that only authorized parties can access it. Hardly any efforts on implementing encryption can be found in photography, which is a remarkable finding, keeping in mind that photographs are generally among the most private media. This statement becomes even more astonishing considering the year, when the first patent for such a solution was registered: 1999. [6] To fulfill the requirements of privacy in photography it might be necessary to encrypt photographs right after they are taken.

background and motivation Especially in the context of medicine this is very important as health data is considered the most private data. [7] Medical photographs show a patients body, they are used to document visual properties of the skin, the state of a wound or the progress of a plastic surgery. All of the above mentioned photographs document very private information about an individual, which is only to be seen by authorized persons. WHAT ABOUT THEF? MENTION?

Another important application for encryption of photographs is journalism. Journalists are looking for possibilities to hide any traces of sensitive material on their devices, for obvious reasons. This is called “plausible deniability”, or to be more precise, according to the Oxford dictionary, plausible deniability is “the possibility of denying a fact, especially a discreditable action without arousing suspicion”.[8] Journalists take big risks on them when taking precarious pictures in countries with authoritarian governments, where such actions are considered as act of crime and punished with imprisonment or even death penalty [9]. So it could be possible that encryption would not be enough to keep the journalists safe, as a trace of encrypted images could already lead to punishments. Encryption is a first step to achieve this level of privacy. For this reason 150 journalists wrote an open letter to camera manufacturers, asking them to implement such a feature, in December 2016.[10]

description of the addressed scientific problem So we can state that there is a demand for digital single-lens reflex (DSLR) photo encryption feature which is not (yet) satisfied by state of the art devices. This fact may lead physicians or journalists to take their private devices, which support encryption, for taking photographs, which is no real option for neither professional privacy nor professional image quality.

Privacy is strictly regulated by law, the most important regulations may be found in the general protected data regulation of the EU [4], the European convention of human rights [5] or the respective national law. [1, 2, 3]

Another issue that may be stated here is that even if custom solutions for encryption do exist, they are not implemented to be turned on and off again easily. This leads to the conclusion that it is not possible to perform image encryption without a required amount of previous knowledge and expertise with such solutions.

Our goal is to evaluate different possibilities, to make on-the-fly encryption for DSLR cameras possible. To ensure comparability between the introduced possibilities it is inevitable to define measurable parameters.

As encryption requires computational resources, its performance is dependent on the platform it is used on. Considering this fact it is essential to be aware of the limited hardware resources of DSLRs that can be used for encryption. So we can state that the performance of the implemented algorithm may be measured by the speed of the encryption process ($v_{encryption}$). Furthermore, a threshold must be found that determines the minimal speed (v_{min}) of the encryption process.

Additionally, it must be assured that the encrypted information may only be decrypted by authorized persons. For this reason, the used encryption algorithm must be considered safe, according to the state of the art.

To ensure a specific degree of performance it is necessary that the evaluated solution does rely on as little proprietary soft- and hardware as possible. An important information which is not known to us for now: whether or not it is better to perform encryption on DSLRs with software or hardware. In the first place, this is a performance related topic, but it has big impact on portability.

Our final goal is to create a method which can evolve to a best practice, which gives the medical and journalist community the possibility to protect their data with the help of encryption. A best practice solution should fulfil the following requirements in the best possible way.

1. The speed of the encryption process must be greater than the defined threshold ($v_{encryption} \geq v_{min}$);
2. The used encryption algorithm must be considered safe.
3. The solution must not be limited to one DSLR manufacturer.
4. Plausible deniability must be assured.

explanation of fundamental terms and basic definitions PD

1.1 Encryption

As in this document we are going to write about solutions, which use encryption, we think it would be good to provide the reader some basic information about encryption. According to the Oxford Dictionary encryption is the process of converting information or data into a code with the goal to prevent unauthorized access. [11]

In the field of cryptography there are two ways to encrypt something. Either asymmetric or symmetric encryption algorithms can be used. The main difference in the two methods are, that in case of the asymmetric encryption two keys are beeing used. One for the encryption and a second one for the decryption. On the other hand in case of the symmetric encryption there is onlye one key used both for the de- and encryption. As we, during our project, mostly used symmetric-key schemes, we are going to focus on this, and we will not write about public-key schemes.

There are two major groups of symmetric encryption algorithms. These are the block- and streamciphers. Blockcipher algorithms devide the data into equal size blocks, and with the help of the encryption-key they perform the encryption on each blocks individually. Streamciphers are a special kind of blockcipher, where the blocksize is one byte. [12] Further classification of block- and streamciphers will not be included, as it would go beyond the scope of this paper.

1.1.1 Advanced encryption Standard

The currently used algorithm for encryption is the Advanced encryption standard (AES), which was defined in 2000, to replace the previous one, which were considered unsecure.

AES uses the Rijndael algorithm with some restrictions. Rijndael was invented by Daemen et. al. [13]

As Rijndael is a blockcipher the round transformations will be applied on the datablocks in round number (N_r) of times, which can either be plaintextblocks or cipherblocks. The input for the algorithm is the key and the plaintextblock, and the output is the cipherblock, or the cipherblock with the key, and the output is in this case the plaintextblock. [13]

First the algorithm derives the round key from the userkey, which will be used during the inital key addition. After this, the round tranformations will be applied N_r-1 times, and finally during the final round the encryption considered done. One round consits of four trandformations,

SubBytes — This is the only non-linear transformation of the algorithm. Every byte will be mapped on an other with the help of a Substitution-Box (S-Box).

ShiftRows — During this step every row of the state (which is a 2D-array representation of the block) will be shifted cycliically over different offsets.

MixColumns — In this step every column will be multipied by the following polynome $c(x) = 03 * x^3 + 01 * x^2 + 01 * x + 02 \mod x^4 + 1$

AddRoundkey — In this final step we apply bitwise XOR to the current block with the roundkey.

The decryption uses the inverse of the round transformation, which means they do the same, only in the other direction. [13]

1.1.2 ChaCha

Without beeing too detaild, we are shortly going to introduce the ChaCha encryption algorithm, as we are going to use it later.

ChaCha is a derivate of the Salsa20 algorithm family. The Salsa20 algorithm is a stream cipher, so according to our previous deffinition by Menzenes et al. in [12] we are talking about a cipher where the blocksize is one byte. However according to Bernstein, Salsa20 "generates the stream in 64-byte blocks". [14]

But without beeing cought up too long with the categorization of an algorithm lets jump to the description of how Salsa20 and ChaCha work. Salsa20 uses a 256-bytes key, which will be expanded with a 64-bit unique message number to a 2^{70}-bytes stream. To encrypt b-byte of plaintext it takes the first b-bytes of the stream and xors it with the plaintext. The rest of the stream will not be used. The decryption works just like the encryption. Salsa20 works only with 3 kind of operations on 32-bit words. These are addition of two numbers modulo 2^{32}, xoring two numbers and rotating a number by some constant bits. These three operations can be run on every CPU very efficiently, without needing and dedicated hardware like in the case of AES. [14]

ChaCha, as mentioned before is a derivate of the Salsa20-family. There are three differences:

1. Due to performance effects the order of the words within the output block are permutated.

2. The initial matrix which is used during the encryption and decryption is built in a slightly different way. In the case of Salsa20 the user inputs are scattered within the matrix, wheres in ChaCha these reside in the last row of the matrix.

3. "ChaCha sweeps through the rows in the same order in every round."

[15]

outline of the manuscript

2 Related Work

As it is likely when getting in touch with state of the art technology, the number of related scientific papers seems to be limited. The most important research done on this field that could have been obtained is briefly described in this section, to give readers a short introduction on the existing literature. The most outstanding work related to the topic examined in this state of the art report is titled "A VHDL Architecture for Auto Encrypting SD Cards" and was published by the University of Gothenburg in November 2016 [16]. To summarize, the students working on this master's thesis designed an encrypted SD-card for journalists.

The approach was based on a hardware solution. An SD-Card adapter was designed, that applies to the SD-card standard, in other words, the auto-encrypting SD had the size and shape of a usual SD card. Inside this seemingly normal SD-card there was an encryption hardware module, based around a FPGA and a publicly available intellectual property core, which was used for encryption.

A speciality about this approach is that the design aims the SD-card to be used by journalists, who work in "destabilized areas". By this the authors of the thesis mean countries, where a journalists takes big risks when taking photographs and trying to get these photographs out of the country. This is the point, where plausible deniability takes effect: When a new photograph is saved, it first is encrypted and then hidden on the file system, so no traces of the photographs may be found and the journalists are safe.

The encryption of the data is realised with a ChaCha algorithm that is implemented in the used FPGA. For the encryption a pair of public and private keys is created outside of the SD card. The public key gets stored on the SD card afterwards. The decryption is performed outside of the SD with the use of the private key. It is important here to notice, that previewing the photographs is possible as long as the SD card is powered up.

What is missing in the described approach is a hardware implementation. All of the work that was done to achieve what was achieved in this project was carried out on a simulator. Even though the writers of this thesis achieved all of their goals it is inevitable to thoroughly test this solution as a fully developed hardware platform to proof the applicability of the proposed solution.

An issue that has not been addressed completely in our opinion is the plausible deniability. Even though the proposed solution "hides previously created encrypted files from the camera", it is not assured that this file hiding is sufficient towards forensic analysis.

Another approach to this issue, especially focused on plausible deniability was developed by Adam Skillen and Mohammad Mannan at the Concordia University in Montreal, Canada [17]. This solution is basically an app ("Mobiflage") that stores photographs on a deniable file system, which means hiding encrypted volumes within random data on the external storage of a mobile device. The paper was published in 2013.

Google's Android operating system was used for the prototypic implementation of Mobiflage. A precondition for the application of Mobiflage is the existence of an external

memory (SD-card), as the proposed solution does not support internal memory. Two separate volumes are created on the volumes of the SD-card:

1. An userdata volume, used for settings and application data

2. An auxillary volume, where the photos will be persisted

Mobiflage achieves plausible deniability through plausible deniable encryption (PDE) this technique enables the output of different reasonable and innocuous plaintexts from a given ciphertext. The original plaintext (or the image data) is only revealed if the correct key is entered. This makes the decrypted data seem correct for unauthorized individuals trying to force the disclosure of the encrypted information. As Skillen et. al states, there is a file system for Linux called "Rubberhose" which features PDE. Mobiflage features two modes for saving images:

1. Standard Mode In this mode the so called "decoy" password is entered at boot time and non-sensitive data is displayed. The storage medium is mounted as Mobiflage isn't installed.

2. PDE mode In the PDE mode the user enters the "trust" password at boot time and the hidden data may be accessed on the SD-card.

The most important aspect here is that the source of the image is a smartphone or tablet and not a DSLR or any other kind of camera. This fact reduces the time from the capture of the image to the secure persistence on the memory drastically. From the other point of view it might not be sufficient enough for a journalist to take pictures with a smartphone, instead he relies on professional equipment like a DSLR to achieve the desired quality of work.

What is missing in this approach is the universality of the solution. The clear advantage of the approach of Davidsson et. al [16] is that the SD-card may be used in any device, regardless of DSLR or smartphone.

3 Method

The aim of this chapter is to describe the methodology used when performing the research for related literature on the topic of this paper. Several search strategies have been used in order to get as much relevant literature as possible. Relevant literature includes all kinds of scientific papers, conference proceedings, master's and bachelor's thesis covering topics around encrypting and hiding photographs.

A constraint that was applied is that the papers, shall not be older than XXX, as encryption related topics should always be investigated up-to-date, for obvious reasons. The most handy tool in order to filter out relevant literature was Google scholar. Also Institute of Electrical and Electronics Engineers (IEEE) Xplore turned out to be a good and reliable source for state of the art literature in the selected field. To satisfy the medical aspect of the problem statement it was inevitable to use the pubmed database search engine of the National Center for Biotechnology Information (NCBI). Furthermore, the CatalogPlus of the TU Wien, turned out to be a reliable tool when it comes to filtering out scientific valueable content. The following keywords were used in order to get relevant search results:

1. **encryption, cryptography, crypto** This keywords were used to assure that the topic of the found literature correlates to aspects of cryptography.

2. **medicine, medical** As the search results may include application in the field of medicine these keywords can be used.

3. **journalist, journalism** To receive search results that stand in relation to journalism these keywords may be used.

4. **camera, digital camera, dslr** As lots of papers discuss the encryption of images on convenient PCs, these keywords are necessary to filter for the application of cryptography on such devices.

In the context of medicine it was important to filter out results that are only related to medical imaging techniques like MRI. This is not because this research is limited to the content of the image recorded, but rather to the device, the image was captured with. For this reason, papers addressing issues in dermatology or wound management were considered relevant as these domains tend to use conventional image capturing devices.

Numerous papers were found regarding the issue of watermarking. Watermarking, in this context, describes the process of modifying an image so it is possible to identify source, creator or owner of a photograph. So we can state, that watermarking addresses issues in the field of integrity but does not provide any advantage in terms of confidentiality, as it is required by the topic of this paper.

The result of the above mentioned research is a set of relevant literature that marks the state of the art in the topic of this very scientific work. All in all it was possible to obtain number papers. List papers somehow, but maybe not in this chapter?

describe how you arrived at the set of techniques presented in the Results section which journals / conference proceedings / libraries / digital libraries / search engines have been searched? what keywords have been used? how many results were obtained? what were the selection criteria? how were the relevant ones selected? which ones were excluded? why? were there any other constraints (e.g., year of publication, particular application area, etc.)?

4 Results

At this point, we have obtained all the papers that are valuable for our research. In the following sections, the results of our research are discussed and even more importantly, compared to each other. The state of the art in the selected topic is described by the entirety of the gained knowledge that is compressed in these results. Furthermore, we will give a short overview of our comparison at the end of this chapter.

SUMMARY The most outstanding work related to the topic examined in this state of the art report is titled "A VHDL Architecture for Auto Encrypting SD Cards" and was published by the University of Gothenburg in November 2016 [16]. To summarize, the students working on this master's thesis designed an encrypted SD-card for journalists. The approach was based on a hardware solution. An SD-Card adapter was designed, that applies to the SD-card standard, in other words, the auto-encrypting SD had the size and shape of a usual SD card. Inside this seemingly normal SD-card there was an encryption hardware module, based around a FPGA and a publicly available intellectual property core, which was used for encryption.

Another approach to this issue, especially focused on plausible deniability was developed by Adam Skillen and Mohammad Mannan at the Concordia University in Montreal, Canada [17]. This solution is basically an app ("Mobiflage") that stores photographs on a deniable file system, which means hiding encrypted volumes within random data on the external storage of a mobile device. The paper was published in 2013. Google's Android operating system was used for the prototypic implementation of Mobiflage.

Similarly to Mobiflage another paper was identified that addresses the issue of secure storage of photographs on mobile devices. Landman et. al [18] published a paper called "A Mobile App for Securely Capturing and Transferring Clinical Images to the Electronic Health Record: Description and Preliminary Usability Study" that describes an app developed for iOS devices called "CliniCam". To enhance security, this app does not save the image permanently, it rather transfers it to a secure electronic health record (EHR) in PDF format. The temporary storage of the photographs is handled within "a secure temporary storage area allocated to the app" this storage area is called sandbox. SUMMARY

4.1 Speed

A critical factor for the quality of the solution is the speed of the encryption process. This is the time between the capture and the secure storage of the image. Images in the so-called RAW-format are the purest way of storing images, as the loss of quality through compression is minimized. The size of a RAW photograph is highly depending on the quality of the used camera but is usually between 20MB and 50MB. As already stated in the introduction the speed of the encryption process must be greater than the defined threshold ($v_{encryption} \geq v_{min}$). Davidsson et. al [16] state in their paper that the transfer speed shall be at least 10MB/s. As no other reliable source for the minimal speed requirement could have been found, this 10MB/s are considered state of the art. As a results we can state that v_{min} is 10MB/s.

We start with the analysis the approach of Davidsson et. al. In their paper three bottlenecks are stated that define the speed of the encryption process. The first one, the processor operation is stated to not affect the speed of the data transfer. Secondly the encryption reaches a speed of 20MB/s and is therefore non critical for the success of the implementation. Finally, the speed of the data transfer to the SD-card is mentioned, which does not meet our speed requirement. This is caused by the used SPI interface, which only supports a maximum speed of MB??? http://ieeexplore.ieee.org/abstract/document/4762946/.

Mobiflage, the solution described by Skillen et. al [17] is hardware dependent. The performance tests described in the paper were carried out on a Nexus S smartphone running Android 4. The test conditions were as follows: Four files with sizes between 50MB and 200MB were written to the encrypted memory. This process was repeated 20 times per file, resulting in an average writing speed of 5288±69MB/s. Unfortunately, this value only reaches 50% of the desired speed.

The analysis for CliniCam [18] is quite complex as the performance is not described in the paper. As the image is not directly encrypted but rather just stored in the "secure storage area" and the storage is a highly performing NAND-flash, we can guess that the storage is not a bottleneck of the application. As the image is not aimed to remain on the device, a transfer to the EHR server via WiFi is necessary. The issue here is that this process must be started manually, which decreases the speed of the overall operation. If a device with the 802.11ac stanard is used, a speed of arround 35MB/s may be reached, which is sufficient for our requirement.

4.2 Universality

As we can state for now there are plenty of possibilities to reach the target of professional photo encryption. An issue that has to be aimed now is, whether the identified approaches are can easily be used across all kinds of use cases or not. For this reason we will now take a look at the universality of the relevant attempts.

Davidsson et. al [16] designed a solution that is based on a hardware platform. This hardware platform is designed as SD-Card, not only in terms of size and shape this SD-

Card is capable of communicating with all kinds of devices using an Serial Peripheral Interface (SPI) Bus. So theoretically it would be possible to use this SD-Card in any digital camera, DSLR, PC or smartphone, that supports SD-cards. A big problem here is the design as SD-Card, as the standard specification for smartphones is to use micro SD-Cards in order to keep this devices compact.

The approach of Skillen et. al [17] is the complete opposite in terms of design. As this solution is designed as an Android app, it is of course limited to Android smartphones, with at least one micro SD-Card slot. Considering that Android is the prevailing operating system for mobile devices Mobiflage is a promising application in terms of universality, at least for non-professional usage. Anyway, porting this application for usage on professional DSLR devices will nearly be impossible unless the DSLR is powered by an Android powered operating system.

Similarly as the approach of Skillen, Landman et. al [18] deal with an approach that shows disadvantages in terms of universality. This is caused by the single-platform design of the app CliniCam that allows it only to be executed on iOS platforms. Another issue is that the platform iOS is bound to a single manufacturer, which makes CliniCam even more limiting.

OVERVIEW/TABLE: WHICH APPROACH IS THE BEST

4.3 Security

As one of the crypto-algorithm which was used by us is by the NIST standardized, and still unbroken AES, and the other one is also a well tested algorithm, currently there should be no security issue regarding the cryptological strength of the used algorithm. However for the sake of completeness, we will describe some key aspects of the security of the different solutions.

Rijandael was chosen by the NIST to be the new encryption standard, back in 2001. Although some modes of encryption (like the ECB mode for more than one block) are considered week and insecure, the security which is provided by the algorithm remains unbroken.

As what Salsa20 and ChaCha consideres, Salsa20 was one of the finalists of the ECRYPT Stream Cipher Project. [14] Several papers reported, that they managed to break the encryption of Salsa20 or ChaCha if only 8 or less rounds were used. [19, 20, 21, 22] But both ChaCha and Salsa20 are capable to work with up to twenty rounds, which even after twelve years after its submission to eSTREAM remains unbroken.

Keeping all this in mind, we can say, that the VHDL solution by Davidsson et. al. is considered secure, as they use in their implementation ChaCha20 (this means ChaCha with 20 rounds). [16]

Skillen et. al. uses in their solution AES in XTS mode. XTS mode is commonly used for full disc encryption, and is also considered secure. [23]

The CliniCam paper provides no specific information about what kind of technology is beeing used. The authors of the paper only say, that

a "The transmitted images are signed with the unique secret key of the user." This indicates, that some kind of asymmetric crypto algorithm is used.

b "The images are stored temporarly in an encrypted area of the application memory." This might indicate some symmetric crypto, however it would be interesting to know, where the key to this area resides. It is probably also in the memory, outside of the encrypted area, in which case the encrypted area does little help.

c "Secure wireless transmission" The authors mention, that the transmission is only allowed through the secured WiFi network of the hospital. However they don't provide any information what kind of technology is used in this case, like if it is WEP or WPA/WPA2 or something else.

All in all we categorized CliniCam in the point of security "unknown", as the authors didn't provide any specifics regarding the used technologies. [18]

4.4 Plausible deniability

Used Keywords: deniable encryption, plausible deniability sd card

Denying the unauthorized access to information is relatively easy to achieve with encryption, encryption still doesn't solve the problem to hide the existence of the encrypted data. As prevously have been showed with choosing a strong enough key, cryptographic algorithms can act as a feasable protection, however, in some states the legistlation can oblige the individuals to disclose their keys. This means an advisory can often observe, that somebody tried to hide some information with the help of encryption, and maybe this advisory can even see the size or some other metadata of the file. As it was reported in the news the former director of the NSA and CIA said, that they kill people based on metadata. [24] It is worth mentioning, that plausible deniability is only important in the context of journalism and not in the field of medicine.

To acchiev plausible deniability we found two methods. Either we solve the problem on the file system level, which mean, we take a file system that allows to be configured in a way, where plausible deniability is possible, or we use deniable encryption.

For the file system we found the by Peters et. al. presented DEFY-file system solution. DEFY stands for **D**eniable **E**ncrypted **F**ile System for **Y**AFFS. YAFFS stands for **Y**et **A**nother **F**lash **F**ile **S**ystem. The basic idea for DEFY is, that it defines different deniable layers, which have to be opened separatly. And as the whole file system is encrypted, an advisory can't find out if changes on the file system originate from some normal OS activity, or that the user tries to hide something. Additionally they make sure, that if something gets deleted it is going to be earsed from the drive. [25]

The basic idea behind plausible encryption is, if the advisory aquires the ciphertext and obliges the user to hand over appropriate key, then the user can hand over a fake key,

which will lead to a biased decryption, which doesn't include any sensitive data, however the advisory can't tell if the decrypted information is fake or real. [26] The problem is not that, that the decryption algorithm can't decrypt an information with a wrong key, because there are ways to configure the system in a way, that the algorithm doesn't check if the decrypted information is valid or not. The problem is, like Trachtenberg describes, to create an output with a fake key, that will convince the advisory, that he got the real output. [27] Trachtenberg in [27] later describes a method how to implement deniable encryption based on the *set reconciliation* in the context of strings. However in our case this method is not really applicable, thus we have to handle specifically designed file formats, which even after decryption with a fake key should deliver a valid looking picture.

Davidsson et. al. doesn't mention if there would be an option in their case to apply methods which would enable plausible deniability. The same applies to the solution by Skillen et. al.

5 Discussion

6 Conclusion

References

[1] D. Bundestag, "Gesetz für sichere digitale kommunikation und anwendungen im gesundheitswesen sowie zur änderung weiterer gesetze," 2015. Version 06.04.2017.

[2] O. Nationalrat, "Datenschutgesetz 2000," 1999. Version 06.04.2017.

[3] O. Nationalrat, "Gesundheitstelematikgesetz 2012," 2012. Version 06.04.2017.

[4] E. Parliament and E. Council, "General data protection regulation," 2016.

[5] E. C. of Human Rights, "European convention on human rights," 2010.

[6] E. Steinberg and V. Yeremenko, "Method and apparatus for in-camera encryption," Jan. 19 1999. US Patent 5,862,217.

[7] P. Williams and E. Hossack, "It will never happen to us: the likelihood and impact of privacy breaches on health data in australia.," in *Health Informatics: Digital Health Service Delivery - The Future is Now*, pp. 155–161, 2013.

[8] "plausible deniability - oxford dictionaries." https://en.oxforddictionaries.com/definition/plausible_deniability. Online; Accessed: 30.03.2017.

[9] A. International, "Egyptian photojournalist at risk of death penalty." https://www.amnesty.org/en/get-involved/take-action/journalism-is-not-a-crime-free-shawkan/, 2016. Online; Accessed: 06.04.2017.

[10] F. of the Press Foundation, "Camera encryption letter." Open Letter to Camera Manufacturers, December 2016.

[11] "encrypt - oxford dictionaries." https://en.oxforddictionaries.com/definition/encrypt. Online; Accessed: 07.05.2017.

[12] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 1996.

[13] J. Daemen and V. Rijmen, *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, 2013.

[14] D. J. Bernstein, "The salsa20 family of stream ciphers," in *New stream cipher designs*, pp. 84–97, Springer, 2008.

[15] D. J. Bernstein, "Chacha, a variant of salsa20," in *Workshop Record of SASC*, vol. 8, 2008.

[16] A. Davidsson and T. Rasmusson, "A vhdl architecture for auto encrypting sd cards," Master's thesis, 2016. 36.

[17] A. Skillen and M. Mannan, "On implementing deniable storage encryption for mobile devices," 2013.

[18] A. Landman, S. Emani, N. Carlile, D. I. Rosenthal, S. Semakov, D. J. Pallin, and E. G. Poon, "A mobile app for securely capturing and transferring clinical images to the electronic health record: description and preliminary usability study," *JMIR Mhealth Uhealth*, vol. 3, p. e1, Jan 2015.

[19] J.-P. Aumasson, S. Fischer, S. Khazaei, W. Meier, and C. Rechberger, "New features of latin dances: analysis of salsa, chacha, and rumba," in *International Workshop on Fast Software Encryption*, pp. 470–488, Springer, 2008.

[20] P. Crowley, "Truncated differential cryptanalysis of five rounds of salsa20," *The State of the Art of Stream Ciphers SASC*, vol. 2006, pp. 198–202, 2006.

[21] S. Fischer, W. Meier, C. Berbain, J.-F. Biasse, and M. J. Robshaw, "Non-randomness in estream candidates salsa20 and tsc-4," in *International Conference on Cryptology in India*, pp. 2–16, Springer, 2006.

[22] Y. Tsunoo, T. Saito, H. Kubo, T. Suzaki, and H. Nakashima, "Differential cryptanalysis of salsa20/8," in *Workshop Record of SASC*, 2007.

[23] M. A. Alomari, K. Samsudin, and A. R. Ramli, "Implementation of a parallel xts encryption mode of operation," *Indian Journal of Science and Technology*, vol. 7, no. 11, pp. 1813–1819, 2014.

[24] "David cole - "we kill people based on metadata"." http://www.nybooks.com/daily/2014/05/10/we-kill-people-based-metadata/. Online; Accessed: 11.05.2017.

[25] T. M. Peters, M. A. Gondree, and Z. N. Peterson, "Defy: A deniable, encrypted file system for log-structured storage," 2015.

[26] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky, "Deniable encryption," in *Annual International Cryptology Conference*, pp. 90–104, Springer, 1997.

[27] A. Trachtenberg, "Say it ain't so-an implementation of deniable encryption,"