Military Technical College
Kobry El-Kobbah,
Cairo, Egypt

11th International Conference
on Electrical Engineering
ICEENG 2018

C2OS, A New Cryptographic Operating System for Smart Cards

Hazem M. Eldeeb*, Hisham M. Dahshan,* and Alaa El-Din R. Shehata*

ABSTRACT

Smart card is a miniature computer with very limited hardware and software resources. Like any computer, an operating system is needed to manage the card hardware and software resources. Several smart card operating systems of different types were developed for this purpose. The basic functions of these operating systems are: handling the card input/out process, managing the file system, managing communication with card users/ application programs and data exchange with the cryptographic algorithms embedded in the card, if any. The user/application is allowed to interact with cryptographic algorithms with their default parameters and with no possibility of cryptographic parameters customization. This paper aims to make the smart card smarter by presenting a new type of smart card operating system that covers a new area of commands. The new set of cryptographic commands enables the applications/developer to deeply access the cryptographic primitives and customize their building blocks at run time. In order to test the new command set and demonstrate its features, the new operating system has been developed in embedded C language and implemented on an open platform card.

KEY WORDS

Smart Card, Operating System, Cryptographic Primitives.

1. Introduction

Today, smart cards are widely used in our daily life. Their technology is being used in many fields like: credit cards, passports, health cards, ID cards, driving licenses, SIM cards for mobile phones, etc. Smart cards are originally known as integrated circuit cards (IC cards). The reason for naming IC cards with smart cards is that the card functions are not limited to those functions defined only at build time. The set of card functions could be extended in run time according to the system they work in and also according to user requirements. Smart cards with processor chip need an operating system known as Card Operating System (COS). The basic functions of COS are: managing the card resources, and enabling instructions execution and communication with the outer world. A variety of card operating systems have been developed.

* Egyptian Armed Forces.

They vary from single application to multi-application operating systems and from single user to multi user operating system, and also from closed to open architecture operating systems. Another important function of card operating systems is data exchange with the cryptographic algorithms (primitives) programmed on the card, if any. These cryptographic primitives (symmetric-asymmetric) are statically programmed on the card at programming time. It is used as is by the card holder at run time and with no possibility of modification or customization.

1.1 Smart card Components

A smart card is a very compact, portable and pocket-sized computer. The card holds one or more embedded chips according to the application(s) it is designed for [1]. These chips could be memory chips (usually a serial EEPROM) and/or CPU (for example a PIC or AVR CPU). Besides the chip(s) and the card itself, there is also a contact area where the outside world can contact the chips. The contact area may have the shape of square or oval area of 6 or 8 contacts and usually coloured in gold as shown in Fig. 1. The connections should comply with the standard ISO78162.

1.2 Smart Card Classifications

In general, smart cards are classified according to the contact method or according to the type of the embedded chip inside.

1.2.1 According to the contact method:

- Contact cards: in this type of cards~~,~~ –, there is a small metal pad (~~Electrical~~ electrical contacts) located on the outside of the card to enable connection to the outside world. The card must be inserted in a card reader (Card ~~acceptance~~ Acceptance ~~device~~ Device CAD) to make a physical connection between its connectors and the reader connectors. In addition to input/output ports, this connection provides the card with the power and clock signal required for the chip to operate. The card power is switched on or off under the control of the host application or the reader (ISO 7816 Part 1 standard).
- Contactless (proximity) cards: in this type of cards, a chip exists inside the card but with no physical connection between the card and the reader. The card is passed along the exterior of the reader or several centimetres away and use radio frequency (RFID) and –very tiny embedded antenna for signalling and communication with card reader (ISO 10536 and ISO 14443) [2].

1.2.2 According to the type of the embedded chip:

- Memory cards: ~~In~~ in this type of cards as shown in Fig. 2(a), the card does not contain a microprocessor; it directly stores data with no processing. Some of these cards are memory-protected, they have a built-in security logic to control access to the memory, and this is normally done using a PIN code. Memory cards are suitable for low-cost use like: prepaid cards, pay phones, car parking and vending machines. Memory cards are considered passive devices because they store data only without any processing.
- Processor cards: they are smart cards with embedded microprocessor, ROM, RAM and some peripherals as shown in Fig. 2(b). These cards are capable of performing on-board processing for the data stored or received from the card

reader. Normally the processor has a very limited resources and its word size varies from 8-bit to 32-bit.

1.3 Smart Card Communications

~~As mentioned before, t~~Smart card needs a card reader (CAD) to communicate with any user/application. The communication is half-duplex and it is based on a master (reader) and slave (smart card). At the link level, the two most used protocols for data exchange between the card and the reader are ~~Asynchronous~~ asynchronous (single) command/response (T=0) which transmits data per bytes, and ~~Asynchronous~~ asynchronous (multiple) command/response link-level protocols (T=1) which transmits data per blocks [3]. Smart card communications is shown in Fig. 3.

The communication between the card and the reader is done via APDU (Application Protocol Data Units). APDU is a message structure/communication unit between the smart card and the card reader. It carries a command from the reader to the card and brings the response from the card to the reader. An APDU from the reader to the card is called command APDU (C-APDU) , and an APDU from the card to the reader is called response APDU (R-APDU).

1.4 Smart Card Security

One of the most important advantages of smart cards is that they provide tamper-resistant storage for protecting sensitive information and parameters which can be protected against unauthorized access and manipulation. The stored data can be accessed only by the serial interface and under operating system control. Another important feature is that data can be kept encrypted with on-board cryptographic algorithms with cipher keys and sub-keys that never leave the card [3]. ~~Thanks to the~~Because there are different cryptographic algorithms programmed on the card, users can perform different cryptographic operations without relying on untrustworthy and unknown devices, and therefore avoid potential risks.

1.5 Smart Card Operating Systems

In general, an operating system (OS) is a set of instructions resides in the computer hard drive and loaded into the computer memory by a boot program. The basic functions of operating system are management of all the computer resources (hardware and software) and acting as an interface between the user and the computer hardware. Another important function of the operating system is to manage the application programs by providing services through application program interface (API). Users can directly interact with the operating system using command line or a graphical user interface (GUI). All computers and computer-like devices have operating systems, including: tablets, laptops, smartphones, smart cards, etc. Examples of operating systems; Microsoft Windows, Linux and Unix-like operating systems, android, macOS and Smart Card operating systems. Smart card operating system is a set of instructions programmed on the card ROM and executed by the card microprocessor. The basic functions of the instruction set are: managing file system, ~~managing card~~managing card hardware and software resources~~,~~, managing communication with the outer world, ~~and~~ data exchange, and data security management. Smart card operating systems have many classifications. One of the most important classifications is the classification according to applications management.

- Closed (Native)/ dedicated operating systems: in which the operating system includes commands designed for specific applications. These operating systems have a high performance and optimized memory consumption since they are designed and tailored for a specific application and hardware resources.
- General purpose/Open (Global) operating systems: the operating system includes generic command set for general use of the card. These operating systems are designed in a way that allows the developers to use any high-level languages for writing their own applications to interact with the operating system. On the opposite of closed operating systems, the code size of open operating systems is relatively large and memory consumption is not optimized [4].

Smart Card operating system allows the user to interact with the cipher algorithms programmed on the card. The user is obliged to use these algorithms with their default parameters with no possibility of customization.

2. Related Work

Several operating systems of different types were developed. They were developed in different programming languages and on different platforms. The type of the operating system defines the field the smart card works in [5]. MULTOS operating system is an open and Multi-application, multi-purpose smart card operating system that enables a smart card to carry a several applications. It supports variety of programming language (C, Basic, and java), and it works under Java platform [5]. MULTOS supports several cryptographic algorithms like: AES, DES, SHA, 3DES and RSA [6] and with no possibility of customization.

BasicCard operating system is an open, Multi-application, multi-purpose, and ISO/IEC 7816 compatible operating system. It works with Basic programming language, and it works under Basic platform—. BasicCard supports variety of cryptographic algorithms like: AES, DES, SHA, ECC and RSA and with no possibility of cryptographic algorithms customization [7].

Windows operating system is an open, Multi-application, multi-purpose operating system. It is fat-based file system and ISO/IEC 7816 compatible. It works with Microsoft visual Basic and C++ languages under dot Net platform. Windows operating system supports several cryptographic algorithms like AES, DES, SHA, ECC and RSA [8]. CardOS is ISO/IEC 7816 compatible, closed and Multi-application operating system. It has a dynamic and flexible file system, with the possibility of adding new files or programs at run time [9].

Why we don't use those two operating systems? **:[Commented [h1**

3. ~~the~~ The ~~new~~ proposed Cryptographic Card Operating System (C2OS)

The proposed cryptographic card operating system (C2OS) is a set of instructions developed in embedded C language and embedded in the smart card non-volatile memory (ROM) as a binary file. Using an application program interface (API), any application can communicate with the card and configure/—customize the cryptographic primitives programmed on the card and get use of them. ~~As mentioned above, t~~The main advantage of the proposed operating system is that the user with the help of set of commands has an access to the cryptographic components /parameters at run time without altering the main structure of the algorithm. The card can be programmed with the most highly secure algorithms with their default parameters which can be changed by the user at any time, if required.

3.1 Customizable Cryptographic Primitives

The main idea of C2OS is keeping the customizable parameters of the cryptographic primitives in the card programmable non-volatile memory to allow access at run time. For the purpose of illustration, the card has been programmed with two cryptographic primitives; block cipher and shift-register based pseudo-random number generator as shown in Fig. 4. The pseudo-random number generator acts as a stream cipher and random numbers generator.

3.1.1 Block cipher

The The implemented block cipher is 10- rounds, 128-bit balanced Feistel structure block cipher algorithm with cipher key length equals to 256 bit. In general, block cipher is designed in one of two structures, substitution permutation network (SPN) structure or Feistel structure [10]. Due to the limitation of the target smart card resources, and in order to save memory, a block cipher with Feistel structure has been chosen to be embedded on the card. Only one entity of the algorithm is implemented for both encryption and decryption with the same set of sub-keys. The reversibility of the algorithm is independent of the reversibility of the algorithm round function and it is guaranteed only by the Feistel structure itself. The algorithm components include highly non-linear substitution tables and diffusion matrices with maximum distance separable property. In order to allow access at run-time, all the customizable parameters have been kept in the card programmable read-only memory (EPROM). In order to guarantee a high level of security, the pre-set (default) values and parameters of the building components of the algorithm have been chosen carefully with very good cryptographic properties compared to other highly secure standard cipher algorithms like AES. The user has the capability to use the block cipher algorithm embedded in the card with its default values or to customize some basic component values like substituting table, number of rounds, round constants, permutation tables and diffusion matrix.

3.1.2 Pseudo-Random Number Generator

As mentioned before, tThe stream cipher algorithm is based on pseudo-random number generator (PRNG) with an internal state buffer with length equals to 128 bit. Basically, the Pseudopseudo-random number generator is used to generator generate random values with any length. and itIt is also used as a stream ciphers by Xoring its output with the plaintext. The algorithm structure includes linear feedback shift registers with non-sparse feedback polynomials to provide the required high degree of equivalent linear complexity. It also includes some non-linear components to guarantee the randomness of the output. With the pre-set (default) values and parameters of the building components of the algorithm, the algorithm has passed the NIST statistical test with very good results with reasonable value of significance level. Like the block cipher, the user has the capability to use the stream cipher embedded in the card with its default parameters., or toThe user can also customize some basic component's values kept in the programmable read only memory (e.g. the scalar non-linear functions used in feedback, feedback polynomials, the value of the 128-bit internal state buffer and the number of initialization/warming up clocks).
For both the cryptographic primitives block and stream ciphers, and in order to preserve the high level of security for each of them, the application developer is strictly requested to carefully choose/design and test the new cryptographic parameters before changing the default ones.

3.2 Implementation and Test Card

In order to test and experience the proposed operating system, the operating system has been implemented on a processor smart card. The command set has been programmed and wrapped in a dynamic link library (DLL). Using the DLL and its wrapped functions, any application can communicate with a smart card holding the operating system and manage the cryptographic primitives embedded in the card and customize them if required. The user also can directly do the same job by using a set of commands in the command line prompt.

Although most of the available processor smart cards in the market are closed platforms and they are not available for developing a proprietary operating systems or user-defined software applications, some open platform cards available in the market for other uses (e.g. satellite TV) have been chosen as a test implementation platform. These cards are based on different kind of processors (AVR, PIC,etc).

The problem with these cards is the limitation of resources (small word size-low speed-low memory-, ...etc.). An open platform card "Prussian card" has been chosen to implement and test the proposed cryptographic operating system for smart card [REF] on. The chosen card is ATMEL AVR (processor AT90S8515A) and it has the following specs:

- SRam: 512 bytes
- Data EEPROM: 512 bytes
- Max Freq: 8 Mhz
- I/O Ports: 32
- Programmable Watchdog Timer.
- Programmable Code Protection
- Low-Power Idle and Power-Down Modes.
- Programmable Serial UART
- Master/Slave Serial Interface
- External and Internal Interrupt Sources
- On-Chip Analog Comparator

3.3 Code Development

Since that the smart card has no input/output peripherals like the normal computer, the proposed cryptographic operating system code will be split into two parts:

- Smart card side (on-card) which will hold the binary file for the code for all the instruction set embedded in the card (non-volatile memory).
- Computer side (off-card) which will hold all the command (executable files) for the direct interaction with user, and it will hold also dynamic link library (DLL) for interaction with application programs.

At the card side, all the instructions have been developed in embedded C language for Atmel AVR. A cross-platform IDE (e.g. Code::Blocks for windows operating system) that supports multiple compilers including GCC, Clang and AVR has been used to develop the code. The hardware initialization process takes place at the beginning to configure (data direction-initial values) all the processor ports (input/output) and registers.

```
ACSR=0x80
DDRA=0xff;   DDRB=0xff   DDRC=0xff ;   DDRD=0xbf;    // data direction for port A,B,C,D
PORTA=0xff;  PORTB=0xff;  PORTC=0xff;  PORTD=0xbf;   // ports initial values
```

The main function in the code is set in a polling state to continuously receive APDU commands(C-APDU) from the input port and pass it over to the main engine for processing. After processing, the results is are framed as (R-APDU) and sent to the

output port followed by the status words. In order to save memory, all the constant values and constant look-up tables are kept in the internal system flash (ISF) using the command "PROGMEM". This command can be used to keep the data in the flash memory at the programming time only. In order to enable the user to customize the parameters of the cryptographic primitives at the run time, all the customizable parameters (substitution tables-diffusion matrices-round constants-number of rounds-,.... etc.) are stored in the EEPROM using the command "EEMEM", ". this This command saves the data in the erasable programmable memory.

At the computer side, a set of commands has been developed to allow the user to interact with operating system through the command line prompt. Using any development IDE (e.g. Microsoft visual studio), the set of commands have been developed in C language as separate files. Each command can interact with card inserted in the card reader (CAD) via the standard framework PC/SC.

PC/SC (Personal computer/Smart Card) is an ISO 7816 standard framework for Smart Card access for both Windows and Linux (PC/SC lite) Platforms. This framework allows the interaction between the smart cards and personal computers and it is available in many programming languages like C, C++, and Java. The smart card reader must comply with the PC/SC standard [3]. Fig.ures5, and Fig.ures6 illustrate sample screenshots for commands execution using command line.

3.4 The Command Set

The command set of the new proposed operating system C2OS are divided into three groups; card command group, block cipher command group and PRNG command group. These commands could be used by the user directly in the command line or by the application through DLL. The following tTables 1, 2, and 3s include some samples of the command set of the proposed C2OS. Each record in the table represents the command syntax/function signature and its corresponding APDU data unit. Some functions require more than one APDU command to be executed, and this is due to the limitation of the maximum number of data bytes carried by the APDU[11].

4. Conclusion

In this paper, a new type of smart card operating systems has been presented. The main advantage of the new proposed operating system is that the user has the capability to manage and configure the cryptographic primitives embedded in the card at run time and without programming tools. Using a new set of cryptographic commands, the user has the capability to configure, change and customize the parameters of the cryptographic primitives. which These primitives include substitution tables, diffusion arrays, feedback polynomials, ...etc. The proposed operating system has been developed and embedded on an open platform smart card with 8-bit AVR processor to demonstrate the new added features. Using a smart card with higher resources, the proposed cryptographic operating system could be upgraded and extended to include more cryptographic primitives like; keyed hash function, different block ciphers with different structure (e.g. AES algorithm), stream cipher with different structure, public key scheme, and elliptic curve-based schemes.

Figures

Fig.1. Smart card shape &Pinout

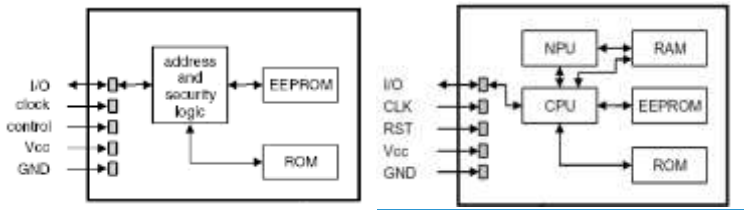
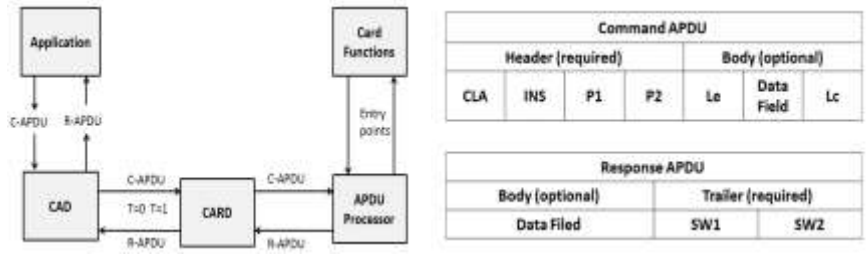


Fig.2. Cards with embedded chip. (a) ~~Processor cards~~ Memory cards (b) Processor cards



~~Fig. 3: Smart card communications &APDU frame.~~

CLA : class of instruction. **INS** : Instruction code.
P1 : instruction parameter 1. **P2** : instruction parameter 2.
Lc : Number of bytes in the data field of the command. **Le** : Maximum number of bytes expected.
Sw1 : Command processing status. **Sw2** : Command processing qualifier.

Fig.3. Smart card communications &APDU frame.

Formatted: Centered

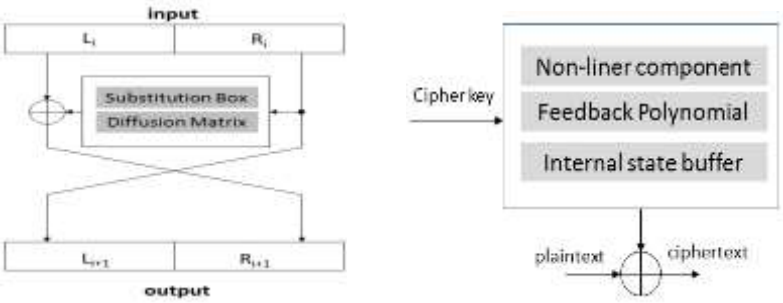


Fig.4.Customizable Primitives. (a) Block cipher (b) PRNG.

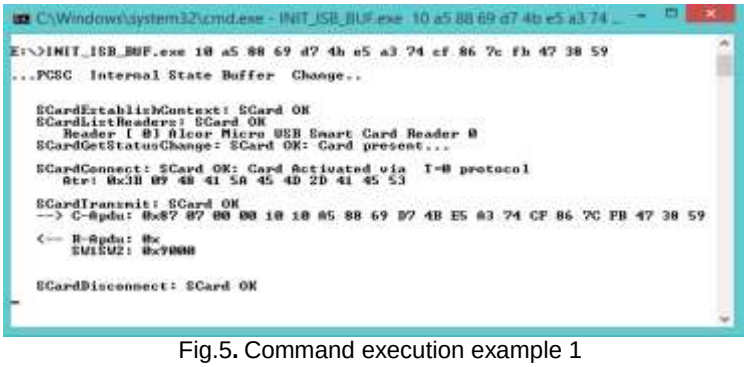


Fig.5. Command execution example 1

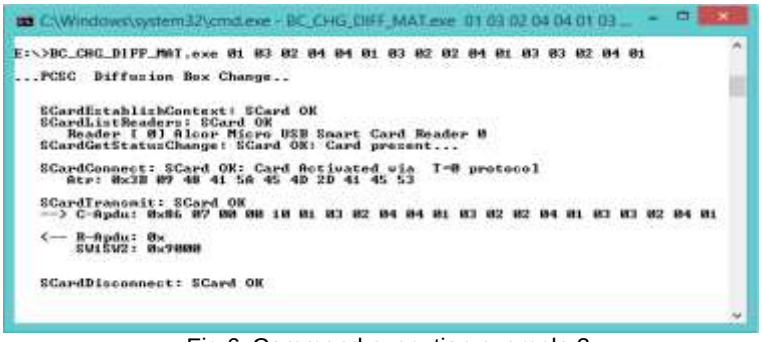


Fig.6. Command execution example 2

Tables

~~Figures 5, 6 illustrate sample screenshots for commands execution using command line.~~

Table 1. Block cipher commands

Command	Function	APDU (bytes)
BC_Enc_ECB(block)	Block encryption in block cipher ECB mode	86 02 00 00 10 x0 .. .. x15
BC_DEC_ECB(block)	Block decryption in block cipher ECB mode	86 03 00 00 10 x0 .. .. x15

BC_Enc_CBC(block)	Block encryption in block cipher CBC mode	86 04 00 00 10 x0 .. .. x15
BC_Enc_CBC(block)	decryption in block cipher CBC mode	86 05 00 00 10 x0 .. .. x15
BC_CHG_SBOX(hex value)	Change the default block cipher sbox	86 06 01 00 ff xx..xx 86 06 02 00 ff xx..xx 86 06 03 00 02 xx..xx
BC_CHG_DIFF_MAT(value)	Change the default cipher MDS	86 07 00 00 ll x1 .. xl
BC_SET_NROUNDS (i value)	Set the number of rounds for the block cipher	86 08 00 00 01 xx
BC_CHG_KEY (hex value)	Change the cipher key for the block cipher	86 09 00 00 LL x1 ... xL
BC_CHG_MAS_KEY (hex value)	Change the cipher master key for the block cipher	86 e7 00 00 LL x1 ... xL

Table 2. PRNG commands

Command	**Function**	**APDU (bytes)**
RND_GET (int length)	Get a random value with specific length (maximum 255 byte)	87 01 00 00 ff
RND_SET_SEED (int value)	Set the PRNG initial value	87 02 00 00 LL x1 xL
SC_LIST_PARAMETERS	List the stream cipher parameters	87 03 01 00 ff xx..xx 87 03 02 00 ff xx..xx 87 03 03 00 02 xx..xx 87 03 04 00 10 xx..xx 87 03 05 00 01 xx..xx
SC_XORT (hex value)	Stream of bytes encryption with stream cipher	87 04 00 00 10 x0 .. .. x15
SC_CHG_KEY (hex value)	Change the cipher key for the stream cipher	87 05 00 00 10 x0 .. x15
SC_CHG_FB_POLY)hex value(	Change the cipher key for the stream cipher	87 06 00 00 ff xx xx xx xx xx xx xx
INIT_ISB_BUF (hex value)	initialize the internal state buffer for the stream cipher	87 07 00 00 LL x1 xL
SC_SET_WAMNUM (value)	Set the number of warm up clocks	87 08 00 00 01 xx

Table 3. Card commands

Command	**Function**	**APDU (bytes)**
GET_SERIAL_NO	Get the card serial number	85 00 00 00 0f
GET_ATR	Cold reset to get ATR code	85 01 00 00 Reset the card
PIN_INTRO	Introduce the PIN code	85 02 00 00 04 00 00 00 00
PIN_ CHG	Change the PIN code	85 03 00 00 08 xx xx xx xx yy yy yy yy
PUK_INTRO	Unblock the card with PUK code and new PIN	85 04 00 00 0c xx xx xx xx yy yy yy yy zz zz zz zz

CHG_PARAM (value, position)	Write a value at certain position	85 05 pp 00 ll x1 .. xl

REFERENCES

[1] G. Selimis, A. Fournaris, G. Kostopoulos, and O. Koufopavlou, "Software and Hardware Issues in Smart Card Technology," *COMMUNICATIONS SURVEYS and TUTORIALS. IEEE*, vol. 11, pp. 143–152, 2009.

[2] Wolfgang Rankl and Wolfgang Effing, Smart Card Handbook, 4th ed., Munich, Germany: John Wiley & Sons Ltd, 2010.

[3] Wolfgang Rankl, Smart Card Applications (Design Models for using and programming smart card). Munich, Germany: John Wiley & Sons Ltd, 2007.

[4] Reza Asgari, Reza Ebrahimi Atan, "Classification of Smart Card Operating Systems," *Computer Engineering and Applications Journal*, vol 3. 11, No 1, 2014.

[5] P. A. Karger, S. McIntosh, E. Palmer, D. Toll, and S. Weber, "Lessons Learned Building the Caernarvon High-Assurance Operating System", Security & Privacy, *IEEE*, Vol. 1, pp. 22-30, 2011.

[6] "MULTOS Standard C-API", Technical Report, MAOSCO Inc, 2013.

[7] (2012) Professional and MultiApplication BasicCard, Product Datasheet, BasicCard Corporation. [Online]. Available: http://www.BasicCard.com.

[8] (2017) Introduction to Windows for Smart Cards, Microsoft Corporation. [Online]. ~~Available~~available: http://technet.microsoft.com/en-us/library/dd277375.aspx.

[9] *CardOS V5.0 Multifunctionality*, Technical Data Sheet, ATOS, July 2012.

[10] William Stallings, Cryptography and Network Security: Principles and Practice, 7th ed., London,UK: Pearson Education, 2016.

[11] Markantonakis, Kostas, Mayes, Keith Tunstall, Michaeland Sauveron and Damien Piper, Smart Card Security: Computational Intelligence in Information Assurance and Security, Berlin, Heidelberg: Springer, 2007.