Smart Cards – Present and Future[1]

Authors:

István Zsolt BERTA – Zoltán Ádám MANN

smartcard@ebizlab.hit.bme.hu

http://ebizlab.hit.bme.hu/smartcard

Supervisor:

Prof. István Vajda

Budapest University of Technology and Economics

Department of Telecommunications

Internet Security and Financial Mathematics Application

Research & Development Laboratory (E-BizLab)

Introduction

Smart cards have been utilized excessively during the last couple of decades. In recent years though, a new generation of smart cards evolved: programmable smart cards. In this paper the authors give an overview of the current state of the technology and compare the cards on the market. They will also examine the trends of development, thus extrapolating their experiences about the present to the future. They also compare the predicted smart card to the ideal one, and try to describe the theoretical and practical boundaries that separate these two.

A bit of history

The ancestor of data storage cards is most probably the calling card. The first **plastic-based card** was issued by Diners Club in 1950. By the end of the fifties two other firms joined the initiation: American Express and Carte Blanche. The first *credit card* was issued by the Bank of America; this is what became VISA later on. Interbank

[1] This paper was issued in **Híradástechnika, Journal on C^5** in December, 2000.

launched another system called Mastercard. However, these early cards were only capable of ’storing’ embossed identification items (names, numbers, codes etc).

The first cards with **magnetic stripes** were developed by the International Air Transportation Association (IATA) in the 1970’s. On this type of card the magnetic stripe stored 210 bit/inch of information, which means about 80 alphanumeric (7-bit) characters. (For the sake of compatibility, today’s magnetic stripes are divided into three regions. The first region corresponds to the original stripe, storing read-only information. The second region can hold additional 40 digits with an information density of 75 bit/inch. The third region is read-writeable and may contain 107 digits.)

A much higher amount of data can be stored on **optical cards**. In this case, both reading and writing (and positioning of course as well) are done optically, enabling a higher level of precision and thus higher information density. Also, typically the whole surface of the card is used for holding data. This way, capacity of some megabytes can be achieved. On the other hand, manufacturing costs of such cards are quite high. Optical cards are used mainly in the medical sector where storage of the patient’s medical records and perhaps even of X-ray photographs is needed.

The next step in the evolution of cards was the appearance of **chipcards** i. e. cards based on the application of microelectronic circuits. The first attempts toward chipcard technology are marked by the establishment of Innovatron in 1974. Bull produced its first card possessing a microprocessor in 1979. On this card, the processor was in a separate chip, which proved to be an insecure solution. However, it was only in the 1980’s that the improvement of the technology made it possible to integrate all circuits on a single chip. In French for instance, chipcards are used since 1986 in public payphones.

Of course the improvement did not stop since then. The first chipcards were **memory cards**. They contained only memory modules which were controlled by the contacts. **Generic cards** were chipcards too. They received instructions from the outside which were processed by the card’s operating system.

Nowadays chipcards possess more and more memory and computation power. This makes it possible that the cards not only execute commands from the outside but also be able to run separate programs. These are **programmable smart cards**. They introduce the concept of a possibly multitasking, multithreaded, multi-user smart card operating system (SCOS). This way, access to the data stored on the card is controlled by the card itself. Thus, the smart card itself can guarantee for security – instead of delegating this responsibility to the possibly untrusted terminal.

Current trends in smart card development

After this short introduction to the history of smart cards it is time to review the currently most important aspects of smart card technology.

Since smart cards are traditionally used to store sensitive data, **security** is one of the most important factors to consider. Security in this case not only assumes the physical tamper-resistance of the card but also the logical integrity and authenticity of the data stored on it. The latter is achieved by the SCOS guaranteeing that the data can be accessed through predefined gates (operations) only. This resembles the object-oriented paradigm in software engineering.

Another approach is the introduction of **files** and **file-systems**. This way, data (possibly belonging to different applications) can be organised into files and directories. Access can be restricted to these files by assigning permissions to them. For each file an access control list (ACL) can be provided defining the rights of the parties (e. g. users, applications) concerning that file. This concept requires **user management**. In this case, a user can authenticate itself toward the card with the help of a personal identification number (PIN) or some other authentication scheme.

As smart cards are becoming smarter and smarter, the usage of more sophisticated **cryptographic protocols** becomes possible. For instance, a traditional data-storage card could be used to store a RSA key. A smart card may also implement the RSA algorithm so it is not necessary to read the key off the card thus improving security.

From the above discussion it should be clear that there exist quite a number of different approaches in smart card technology at the same time. For card issuers **interoperability** is becoming more and more vital. Without international standards the market would be too chaotic for solid applications. This problem was realised in the early stages of smart card history but the process of standardisation is usually slow so the international standards of the field are only former de facto standards made absolute. The most important standard concerning smart cards is **ISO 7816**. Earlier standards – such as ISO 7810 and 7811 – described physical characteristics of different cards. The main significance of ISO 7816 is the specification of the communication protocol for smart cards specifically, although the standard also deals with hybrid cards (e. g. cards with a microprocessor as well as a magnetic stripe).

High-level programming languages are also gaining more and more importance nowadays. Since smart cards possess only very limited resources, one could believe that software development for smart cards is done in machine code. However, this could prove very costly because smart cards from different vendors are usually not compatible on the machine code level. This means that a software manufacturer would be exposed toward the card manufacturer. In such a situation it would be a catastrophe for the software manufacturer if the card manufacturer decided to re-design the card in question or maybe even not to produce it anymore.

Therefore it is an important requirement that programming for the card should be done using a platform-independent, possibly high-level programming language. Of course the need for card-specific optimisation cannot be eliminated but it should be done automatically by a converting tool provided by the card manufacturer. In practical cases studied by the authors the high-level programming language was Java (Java Card API) or Visual Basic although only a subset of the original languages was actually useable. This is not surprising since these languages were designed for use in PCs and smart cards provide a strongly different environment.

Presently people carry 4-5 cards (magnetic and chip cards) in their wallet: ID card, credit card, phone card etc. As the memory of smart cards increases it becomes possible to integrate all these functions on a single card. Running **multiple applications** on a single card would ease our every-day life but would also induce some technical problems. The most important thing is that the applications must not be able to access each other’s data. In this case namely the policeman checking your ID could also access your medical records or have a look at your account’s balance. This criterion puts a heavy burden on the SCOS.

Current trends on the market

With the introduction of generic and programmable smart cards the software market became more and more open. This holds especially for programmable smart cards: even small companies can afford to enter the smart card market as software manufacturers or card issuers because this does not require the expensive equipment of IC technology. Hardware manufacturers on the other hand can focus on producing a couple of card types only but produce them in a large number. This will break down prices and make smart card technology generally cheaper. The importance of this fact should not be underestimated since it is in most cases their currently high price that prevents the widespread usage of programmable smart cards.

As the market extends, interoperability becomes one of the most crucial factors of development. The regulation provided by ISO 7816 did not prove to be enough so the most advanced segments of the market soon declared their own standards (mainly extensions to ISO 7816). Important examples include EMV (Eurocard, MasterCard, Visa) and GSM SIM (Subscriber Information Module). In the evolution of programmable smart cards the PC/SC (Personal Computer/Smart Card) specification plays a similar role.

The **PC/SC Workgroup** was founded in 1996 by leading firms of the PC and smart card market: Bull CP8, Gemplus, Hewlett-Packard, IBM, Microsoft, Schlumberger, Siemens Nixdorf, Sun Microsystems, Toshiba and Verifone. In 1997 they released a specification about the co-operation of smart cards, smart card readers and PCs.

Shortly afterwards, Microsoft released the beta version of the first implementation of the PC/SC specification: Microsoft Smart Card for Windows Professional.

A completely different philosophy is represented by the **Java Card** specification. The Java programming language (developed by Sun Microsystems) is traditionally a high-level, platform-independent, object-oriented language, interpreted by a so-called Java Virtual Machine (JVM). Its platform-independence and widespread usage make it an ideal programming language for the heterogeneous world of programmable smart cards as well but efficiency may suffer damage. In particular, it is not possible to interpret pure Java on a smart card; the byte code of the program must rather be compiled with a card-dependent converter.

The Java Card specification also found quite a number of implementations. DelaRue’s Galactic, Bull’s Odyssey and Schlumberger’s CyberFlex are all Java cards. Nevertheless there are more extreme examples: Java Ring and iButton are also programmed in Java (and conform to the Java Card specification) but have the shape of a ring and a button respectively, instead of the classical rectangular card design. Another example of the Java Card technology is Bull’s SIM Rock’n Tree: a SIM card that can run the same applets as smart cards while it also possesses GSM functions.

Potential purpose of a programmable smart card

Smart cards are small tamper-resistant portable microcomputers. The authors believe that programmable smart cards open a wide range of new possibilities for security-oriented applications. However, they are only used for a few simple aims, and the number of applications which use the benefits provided by the programmable cards is small.

Data storage was the main purpose of magnetic cards and memory cards from the early years. The data stored on the card could be accessed directly from the outside. Generic smart cards provided the first change in this field: the data were protected by an intelligent file system. However, the outside world could still read or write the memory cells of the card. If the PIN codes for the card were compromised, any value could have been written to the card, and this could have left the system in an inconsistent state.

The concept of **object orientation** has been known in the world of software engineers for decades. By the arrival of programmable security-oriented microcomputers, this concept gained a sense in the physical world. A smart card can be treated as an object: it contains data that defines its internal state (attributes), and the data can only be accessed through predefined gateways (methods). The smart card contains both the data and the valid operations on it and they cannot be separated from each other (encapsulation). The data itself is not accessible to the outside world (information hiding) only through the operations. By selecting the appropriate necessary operations and discarding the futile ones, greater security can be achieved. Moreover, the data contained by the card can be kept in a consistent state. Using the object-oriented concept on a smart card, information can be given to a person together with the possible operations. Thus he can be prevented from tampering with the information or misusing it. For example, a letter can be created that destroys itself after the first reading. Or a key that can only be used *n* times to open a door.

The purpose of data storage is often user-identification. A piece of information relevant to a specific user is stored on the card. The system uses this information to identify the user. Three main methods of user identification exist: knowledge based, possession based and biometric. A strong system implements at least two of the three methods independently. In a smart card based system, the second one is already given. In case of generic cards, the reader connects to the card, gains access to the key of the user, and reads it. If this channel is tapped, the user’s secret key is compromised. The same thing happens, if the system’s secret key is compromised: the attacker can gain access to the key in any card. In a system based on programmable smart cards, it can be arranged, that the key cannot be read. No operation exists, that allows the key to leave the secure environment of the card. By taking advantage of the card's computational powers, **challenge and response based identification** can be used. This makes an eavesdropping attacker unable to get the key.

Smart cards are not only data storage devices. They can be blessed with the ability of decision making and thus can be used to protect the rights of not only the system but those of the users too. They make **mutual authentication** possible. When a credit card user inserts his card into an ATM, and supplies his PIN code, he has to trust the machine. If the machine is a false ATM, it may swallow the card (or just keep the card number) and remember the PIN code, and the attackers may drain the credit card user's bank account. It would be preferable, if the card would identify the ATM (or the reader) and check if it possesses the appropriate keys. If the reader fails to identify itself, the program running on the card should send a message to the user to warn him not to supply his PIN code. However, sending warning to the user is only possible through the reader's user interface. The card itself runs using the power supplied by the reader too. This makes the situation a bit complicated, but an algorithm for this problem does exist.

In case of a challenge and response based system, the card can use **encryption** function. Such an algorithm implemented on a card can be used for many purposes. The card can be treated as a machine (or object) that has the following operations:

- Deleting the previous key, and replacing it with a new one
- Encryption using the key in the card
- Decryption using the key in the card

Messages encrypted with the card can only be decrypted by another card containing the same key. The card cannot be copied since the key cannot be acquired from it. Such a card can be used for many purposes. The authors wrote an application for Microsoft Smart Card for Windows, that used the DES algorithm for encryption. They implemented the five pillars of authenticity using it. This card application has another operation: random key generation. If the key was generated by the card and cannot leave it, it makes the card unique. Messages encrypted with a card can only be decrypted by the same card.

This transfers the **criteria of security** from the electronic world (where information can be stolen or copied easily by means not understandable to everyday people) to the physical world we all know well.

A smart card can be used not only to ensure the integrity of a system containing it, but also to **check the integrity** of another system. Considering a PC and a card not as a master-slave architecture but as the interconnection of two computational units, one of them (the card) might be used to check the integrity of the other (the PC). Another PC could also be used to check integrity but we must ensure the integrity of the device we use for checking.

A card could also be used for placing **secure time stamps**. However, for this purpose the card should be equipped with a timer. Dallas Semiconductor's iButton contains a timer so it could be used for this purpose.

Electronic commerce provides a vast area of applications. Such an enormous area is definitely beyond the scope of this article so only some representative examples will be discussed.

Loyalty applications is a wide area where a lot of smart cards are used. By this time mainly generic cards are used for loyalty purposes, but programmable smart cards may open new possibilities for this area too. By defining the proper operations bonus points can be collected for the card in any issuer-defined system, and can be spent in many ways. Another trend is that different companies on different markets combine their loyalty systems and create a common one to gain a higher market share for both of them (e. g: a chain of gas stations and one of supermarkets).

Moreover, cards could be used to pay at electronic warehouses. Either like a credit card, or with digital money. Using a smart card as an **electronic purse** has interesting possibilities. Combining the power of the cryptographic protocols with the object-oriented concept, a card might be used to protect digital cash from copying. In case of digital money copying poses a greater problem than in case of conventional money since making a copy of a series of bits is a trivial operation. This problem does not arise if the software used on both sides is untampered and authentic. Preserving the integrity of software on PCs is practically impossible. A PC's hard disks can be removed, and their contents can be edited on other machines. But in case of smart cards the data cannot be separated from the operations. Thus, when receiving digital money from another card, the receiving card can be sure that the other card does not keep a copy of the money, but deletes it. This assumes that both ends of the transaction are "safe", it is properly encrypted, authentic. This way money can neither be lost, nor generated. "Safe" transactional endpoints can be smart cards equipped with the appropriate keys or certain "trusted services". E. g. Mondex has a similar smart card based electronic purse solution.

Due to its rapidly increasing importance **SIM cards** should be mentioned too. Mobile telephony is the area where perhaps the most of the power of programmable smart cards is used. The external interface of a SIM card is completely the same as that of a smart card and the internal architecture and programming environment is becoming the same nowadays. A mobile phone not only provides a user interface to the smart card and enables knowledge based and biometrical user identification, but also connects the card to a network or to the Internet. There are a couple of cards today that combine the power of smart cards and SIM cards. Such an example is Bull's SIM Rock'n Tree. This card conforms to the Java Card specification, so programs written for Java Cards can be transferred to it. Moreover, it is a SIM card too, and supports GSM instructions.

The ideal smart card

Having examined many of the possible applications of a smart card, the authors now try to collect the possible requirements for the ideal smart card, which would suit all of the needs.

Security was the main purpose of smart cards from the early days. In case of the ideal smart card **tamper-resistance** has utmost priority. The card manufacturer provides the card's physical and the OS's logical security. This is necessary to preserve the application's logical security.

The ideal smart card has **large storage capacity**. It is in the region of megabytes, so even photographs and multimedia information can be stored on it (to be used in e.g. facial recognition). High storage capacity is also necessary to enable the use of more complex applications.

The ideal card is capable of **real-time** speech and video **encryption**. This requires three things: fast computation, fast communication with the outside world and fast cryptographic functions. For the latter purpose it contains a **cryptographic coprocessor** for accelerating both symmetric and asymmetric cryptography.

In case of cryptographic protocols random number generation is vital. This card can **generate good quality random numbers**. If a random number generator can be predicted, whole security protocols can be corrupted. (A good example for this danger is the Fiat-Shamir algorithm, which would be suitable for a smart card due to its relatively low processor requirements on the card side.) It is theoretically proven that the generation of random data is not possible in an algorithmic way. In such cases a human factor is often used. Mobile phones can use disturbances in the ether as random seeds.

The ideal card has an **own power supply** and a **timer**. Equipping a security-oriented microcomputer with timers increases its cryptographic potential. The own power supply enables it to run without the support of a reader. Programmers do not have to keep in mind that the attacker may remove the card from the reader and so cut off its power supply thus trying to leave the card in an inconsistent state. To avoid such problems, **transaction management** should have been implemented. The own power supply not only makes transaction management unnecessary, but provides the possibility for an application to run on the card continuously. Moreover, more applications can run simultaneously in the ideal card's multithreaded environment.

The ideal smart card is a compromise between two philosophies. The upper layer of the software architecture is flexible and replaceable whereas the operating system is strongly connected to the hardware.

The ideal smart card has a **long lifetime** (measured in decades). This, and the fact of storing precious information on the card gives **robust**ness even more importance.

The increase of the number of smart card based applications will definitely increase the number of smart cards held by one person. To solve this problem the ideal smart card runs multiple applications. These may change dynamically so that new applications can be downloaded to the card and deleted when they are not used. It is vital that these applications are separated so that they cannot tamper with each other's data. However, due to lack of storage space and reasons of consistency, it is also vital that they can interact with each other and share data (e. g. cardholder name) or even code.

The card is programmed in a **high-level** platform- (and vendor-) independent **programming language** so that the source code can be easily transferred from one card to another.

The card can be easily programmed because its interface with the standard IT devices is standardized. This not only includes the PC and mobile phones, but communication through the Internet as well.

Trust is critical in case of all security appications. Neither the ideal card's hardware, nor its OS contains any **trapdoors**. Its security features are well documented, and the possible customers can receive information on the smart card's internal architecture. It is not designed using "security by obscurity", but strong cryptography and careful planning and testing.

Developing security oriented applications for PCs is difficult. This is due to their numerous I/O devices, network connections and possible attack points. In case of a smart card the only point where interaction with the outside world is possible is the contacts. The restriction of communications is on one hand convenient, but on the other hand it restricts the power. A simple LED on the card gives the ideal card the ability of **direct feedback** to the user.

Smart card of the future

Now let's examine, what is realistic from the features of the ideal smart card, and what capabilities exist in the cards on the market today.

Today's cards have 8-32 kilobytes of memory. This is likely to increase in the future in parallel with the development of IC technology. Computational power has a closer limit though. Controlling overheating has always been a problem in case of microelectronics but it case of cards the problem is even larger. The card's shape is restricted and plastic may not melt. The authors believe that computational power will increase in the future, but will not increase dramatically. Neither will smart cards' speed nor their storage capacity increase over that of PCs'.

Real-time encryption of speech or video is far beyond the capabilities of today's cards, and the authors believe that it will not be possible in the near future. Supplying cards with cryptographic hardware is a question of price thus it is a question of mass production. Security and portability are the two areas where cards can be better used than PCs. This is why the author suppose that the smart card of the future will be equipped with cryptographic hardware. The production of good quality random numbers is a problem yet to be solved. The documentation of today's cards contains no information on the ways of random number generation.

The possession of an own power supply probably has technical limitations, so cards are not likely to have one in the near future. This implies the absence of a timer too. However, non-card-shaped devices such as iButton do have such possibilities. Transaction management and power supply are two alternatives. On Java Cards the previous one is supported by the programming environment.

Although the Java Card API seems to be a well designed, clear and card-independent programming environment, this part of the technology is still in an infant stage today. Though the language of Java Cards is object oriented, this feature cannot be practically used due to limited hardware resources. This specification has changed a lot in the past and it is likely to change in the future in parallel with the improvement of the hardware.

The lifetime of cards is not measured in decades nowadays. However, iButton is a device designed for hard circumstances (rock climbing, swimming), and its estimated lifetime is much larger than that of smart card. However, the technology of programmable smart cards is new so no long-term experience is available.

The Java Card specification defines an environment, where applets may enter and leave the card dynamically. However, these applets cannot interact with each other. The applications of Microsoft Smart Card for Windows may share data among each other, but the card's filesystem needs to be re-designed before a new application is downloaded. The ideal smart card would have to be a compromise between these two, but security and robustness remain vital. The authors believe that although technology does not have the proper tools for this today, a solution will be found for this problem in the future.

Java Card and Microsoft Smart Card for Windows both use high-level languages today, this seems to be the trend of development. Documentation for the cards is incomplete, the main tool of manufacturers is unfortunately still "security by obscurity".

Smart cards are not likely to have an own user interface in the future. However, their future is closely connected with that of mobile phones. Mobile telephony already gave a boost to the improvement of smart cards, and this rapidly developing area is where programmable smart cards are mostly used in practice today. Combining the power of smart cards with the user interface and network connection of mobile telephones new possibilities may arise. SIM cards already offered the users various applications in the past, and there is more to come.

References:

Cryptographic works

D.W. Davies – W. L. Price: Security for Computer Networks. John Wiley & Sons, 1992.

Bruce Schneier: Applied Cryptography. John Wiley & Sons, 1996.

Gustavus J. Simmons (Szerk.): Contemporary Cryptology. IEEE Press, 1992.

Györfi-Vajda: A hibajavító kódolás és a nyilványos kulcsú titkosítás elemei. Budapest, 1991.

Smart card literature

Berta István Zsolt - Mann Zoltán Ádám: Programozható chipkártyák – elmélet és gyakorlati tapasztalatok (Magyar Távközlés)

Berta István Zsolt – Mann Zoltán Ádám: A hitelesség biztosításának lehetőségei intelligens smart card segítségével (TDK dolgozat)

W. Rankl – W. Effing: Smart Card Handbook. John Wiley & Sons, 1997.

Bruce Schneier – Adam Shostack: Breaking Up Is Hard To Do: Modelling Security Threats for Smart Cards

J. L. Zoreda – J. M. Oton: Smart Cards. Artech House, 1994.

References on the Web

Java Card: http://java.sun.com/products/javacard/htmldoc

PC/SC Workgroup: http://www.smartcardsys.com

Microsoft Smart Card for Windows

http://www.microsoft.com/security/tech/smartcards