Volume 4, Issue 8, August 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Java Smart Card Security Threats and their Countermeasures a Comparative Study

Atul Dhiman
Department of Computer Science
HP University, Shimla, India

Arvind Kalia
Department of Computer Science
HP University, Shimla, India

Anita Ganpati
Department of Computer Science
HP University, Shimla, India

Abstract: Smart Card programming and development began with idea of incorporating an integrated circuit into a plastic card introduced in late 1960s.Smart card numbers were in millions in last decade with multi-application uses i.e. smart cards are used for business transactions and numerous other services. They come integrated with detection and recovery schemes from the security threat problems. This technology can provide identification, authentication, limited data storage and multi-application processing. This paper presents comparative study of threats associated with smart card based on java card technology. These threats are categorized as Logical, Physical and Combined Attacks. Their vulnerabilities with countermeasures have also been discussed.

Keywords: Java Card, Smart Card, Fault Attacks, Logical Threats,JCVM,JCRE, CAP, APDU.

I. INTRODUCTION

Java Cards are multi-application smart cards by Sun microsystems based on theJava technology. The technology makes it possible to operate on a uniquemedium a set of services, called applets that can communicate and co-operate with each other [**3**]. Java card technology combines a subset of the Java programming language with a runtime environments optimized for smart cards and other memory –constrained devices. Same as a size of credit card, a smart card processes and stores information through the electronic circuit embedded in silicon in the plastic substrate of its body. A smart card is a portable and tamper – resistant computer unlike magnetic stripe cards, smart cards also carry information and have processing power [**4**].

With increased usage in varied fields the number of logical and physical threats has also increased. Java cards are reusable and so come with inherent security measures such as applet firewall and byte code verifier.Still there is a range of security threats having damage to most potent level to sensitive information stored.Smart cards despite of vulnerabilities are still used with secure applications for business transactions.

Java Card provides a framework of classes and interfaces that hides the details of the underlying smart card interface and makes it possible to load and run on the same card several applets from different applications [**8**].Java card technology is used to program smart cards and it comprises a subset of the desktop java: a subset of the programming language itself and a cut down version of API [**14**].

Software Architecture of java card technology comprises of two major components**.** First is Java Card Runtime Environment (JCRE)that nicely separates applets from the proprietary technologies of smart card vendors and provides standard system and API interfaces for applets, hence applets are easier to write and are portable on various smart card architectures. Java Card Runtime Environment lies between layer of specialized applets and native smart Card hardware. JCRE consists of java card system component that runs inside the smart card. The JCRE is accounts for following activities:Card Resource Management, Applet Execution, on card system and applet security, Network CommunicationsSecond main component is Java Card virtual machine (JCVM)it executes byte codes, controls memory allocation, manages objects and enforces the runtime security. Java card virtual machine is divided into parts one that runs off card and other that runs on card. Several tasks of processing that are not bound to execute at runtime such as class loading, resolution, byte code verification, and linking optimization are dedicated to the JCVM that is running off card where resources are usually required[**4**].

Off card VMconsists of converter and On Card VM consists of interpreter which together implements VM functions – loading java class files and executing them with a particular set of semantics.

The converter loads and pre-processes the class file that makes up Java package outputs CAP (Converted Applet) file. The CAP file is then loaded on the java smart card and executed by the interpreter. Converter also generates export file representing the public APIs of the packages being converted.

This paper gives an overview of Java card technology and comparative analysis of threats to these java smart cards. The basic architecture of java card has been outlined with different threat categories and their countermeasures are discussed.

II. LITERATURE SURVEY

Several research departments and sections of various licensees are working with sun microsystems on continuous evolution of java card and mitigating with latest threats. This section presents studies related to various card threats and

their countermeasures.WojciechMostowski and Erik Poll in their research paper [**9**] gave the extensive overview of the vulnerabilities and their possible runtime countermeasures against ill typed code with the results of experiments conducted of attacking actual java cards available in market with the ill code. Jip Hogenboom and WojciechMostowski [**5**] presented a simply memory read attack which utilized using type confusion technique to expose faulty transaction mechanism implementation on java cards. Type confusion attack provides application's meta-data, engineer it reversely to get full read and write access. EngelbertHubbers and Erik Poll et.al. [**6**] Investigated the interaction of transaction mechanism with non-atomic methods of Java card API. In results some cards gave unpredictable results if non atomic methods were interrupted with no transaction in process. They concluded that transaction mechanism is the most complex part of the Java Card technology and thus it is difficult to formalize.

Guillaume Barbu and Guillaume Duc et al. (2011[**2**] introduced a new kind of combinational attack on java card 3.0 version and their detailed countermeasures. The new kind of attack has fault attacks and combined attacks taking advantage of an alteration of the operand stack. They provided experimental results proving the practical feasibility of these attacks and giving their efficiency with different approaches to protect the operand stack's integrity.Guillaume Barbu and Christophe Giraud et al [**1**] introduced first Combined Attack on a Java Card targeting the APDU buffer itself, thereby threatening both the security of the platform and of the hosted applications as well as the privacy of the cardholder. This attack leads to an outstanding privilege: accessing the APDU buffer array at any time.Amadou A. SERE and Julien Iguchi-Cartigny [**11**] proposed a set of countermeasures that can be activated by the developer using the annotation mechanism. These countermeasures are efficient and also affordable for the smart card domain, and it is shown by the evaluation of the coverage and memory usage.

III.NEED AND SCOPE OF STUDY

Java Card is an evolving technology it has the ability to support Java applets resulted in a wider variety of applications being supported on a single card. However, the security risks for supporting executable content on smart cards must be carefully evaluated in light of the types of applications the cards will support and multi-application smart card that are in trend in today's era. Multi-application smart cards accept to load code only after an authentication. But in the future, the cards will be open and everybody should be authorized to upload an application. This raises new security problems by creating additional ways to attack Java Cards

Hence, there arises a need for structured explorative study to understand the prevalent and most damaging kind of security issues associated with these multi -application smart cards and to identify possible runtime countermeasures, as these smart cards store sensitive information of the user. This study will help in emphasizing the necessity of taking into account these new threats ,when implementing the latest editions of Java Card and finding the answer to questions that how much vulnerable Java Card applications are to malicious code and how much trustworthy are security mechanisms on the basis of which the java cards are deployed.

IV.RESEARCH METHODOLOGY

Research is a structured enquiry that utilizes acceptable scientific methodology to solve problems and create new knowledge that is generally applicable.Scientific methods consist of systematic observation, classification and interpretation of data. The research methodology to be carried out in this study will include explorative study of literature studying various thesis and research papers. Analysis of the results of the experiments conducted and a theoretic comparison will be made. Finally a tool based approach will be adopted further for the work to be conducted in future which will be concluded later.

V. SECURITY THREATS

Multi-application uses of smart cards have given horizon of opportunities for attackers to access and disrupt the normal functioning of smart cards. Thethreats are classified as follows [**11**]:

Logical Attacks: These attacks exploitthe bugs in the software implementation.

Physical Attacks: These attacks exploitthe analysis or modification of the card hardware.

Combined Attacks: Exploit both physical and logical attacks.

Other Attacks

A. Logical Attack

The three most potent categories of logical attacks are:

Malicious Applet attack: Itis the simplest way to get ill-typed code running on a card is to edit a CAP (Converted Applet) file to introduce a type flaw in the byte code and install it on the card. This will only work for cards without on-card BCV and for unsigned CAP files. For example, to treat a byte array as a short array, it is enough to change a byte load op-code into a short load. Such a misinterpreted array type can potentially lead to accessing other applets' memory and can have potent results.[**9**]

Abusing Shareable Interfaces:Makes use of the shareable interface mechanism of Java Card can be used to create type confusion between applets without directly editing of CAP files [**9**]. To use this to create type confusion, the trick is to let two applets communicate via a shareable interface, but to compile and generate CAP files for the applets using different definitions of the shareable interface defined. It becomes possible because the applets are compiled and loaded separately. This way the server can run malicious code on its own (in terms of context ownership) data [**13**].

Abusing the Transaction Mechanism: The transaction mechanism allows multiple byte code instructions to be an atomic operation, providing a roll-back mechanism if the operation is aborted, for e.g. in case of a card tear. The rollback mechanism should also de allocate any objects allocated during an aborted transaction on a card, and simultaneously reset the references to such objects to null. It is this last aspect which can be abused to create type confusion: if such references are spread around, by assignments to instance fields and local variables, it becomes difficult for the transaction mechanism to keep track of which references should be nullout [**13**].

The further subattacks that can be derived from logical attacks are using: byte as Short Array, object as an array,reference switching with AIDS and reference spoofing.

B. Physical Attacks

Physical attacks are carried out by changing or altering one or other parameter of platform to which the card is exposed. The normal functioning of card chip may be disrupted requiring high end lab equipment. The most potent category of physical attacks is Fault attacks.

Fault attack is an attack which has the ability to disturb the smart card chip by disturbing in physical way his The induced errors can disrupt the computation and can allow an attacker to make changes that he hasn't the privilege to do or can get access to some sensitive data that are on the smart card. Results of faults attacks can be perturbation of the chip registers (like the program counter, the stack pointer), of the memory (variables and code changes) [**12**]. The various kinds of fault attacks prevalent are:

Spike attacks: By varying the power sent to the chip by VCC, it is possible to disrupt a computation.

Glitch attacks:Disturbs the clock speed which can be used to influence conditional jumps by not performing them or performing them when it is unwanted, it results in changes off the program.

Optical attacks: By focusing light with specific wavelengths, it becomes possible to invert the contents of a memory cell.

Electromagnetic Perturbation**:** It is another kind of fault attack performed by creating a strong electromagnetic field near memory thereby disturbing the memory contents to create fine control over the bit need to conduct an attack.

C. Combined Attacks

Combined attacks are harder ones to conduct they involve fault injection followed by code injection to take advantage of values exposed. We here consider Operand stack to carry out such an attack because operand stack is integral part of this frame holding the operands and results of the VM instruction. Most of these instructions consist in popping a finite number of operands, running a specific process and pushing a returned value.

The fault injection aims at the execution of the JCVM. Perturbation can prohibit fully or partially the updating of the operand stack ina push operation. The outputted erroneous value would then be all 0, or all 1.Attacker hence can determine the erroneous values and also can run and attack on his own application on the platform aided with the value previously pushed onto the stack and hence can also control the resulting erroneous value [**2**].After being fault injected following code attacks would be carried out.

Type Confusion Attack: Several instances of application are created of a given class and confused with another class aiming at breaking type safety.

Instance confusion attack: Instance confusion consists in using an instance 'j' of a given class as if it were another instance 'j' of the same class or of a class implementing the same interface. Majority of results of these attack rendered card failure [**2**].

Tampering APDU: This attack leads to an outstanding privilege: accessing the APDU buffer array at any time.

D. Other Attacks

Side Channel Monitoring:Side Channel attacks are physical phenomena to analyse/ manipulate the behaviour of a smart card chip; these attacks are carried out without physically changing or opening the device.

The side-channel is an avenue for acquiring extra information relevant to conducting an attack, in one of several forms [**10**] and these are timing from the execution time of software routines, power, from current owing through on-card circuitry, electromagnetic emanations from current flowing through on card circuitry causing electromagnetic fields. Differential Power analysis and Power Glitch are two modes to conduct side channel attacks.

Non-operational conditions attacks: The idea is to put the card in a state of abnormal operation by varying voltage, frequency of supply and temperature. Attacks on the supply frequency may allow an attacker understanding of the operations performed in the chip combining with processes observation of covert channels.

VI.COUNTERMEASURES FOR JAVA SMART CARD ATTACKS

Each class of attacks have their own countermeasures, some countermeasures are defined in the security policy and are inherently built in the card while manufacturing.

A. Countermeasures for Logical Attacks

Applet Firewall: This feature is inbuilt, prohibits illegal access among applets, Byte code verification: Verifying the byte code produced guards the type correctness of code which in turn assures the memory safety.

The CAP file modification can be prevented by: On Card Byte code verifier (BCV) installation, simply prohibiting applet loading, and controlled applet loading with digital signatures form third party.

Shareable Object interface abuse can be controlled by, making aware client and server of each other .and use of cards with specification preventing use of shareable object interfaces.

Transaction mechanism abuse can be prohibited by: correct implementation of clean-up operation for aborted transactions, or by muting the transaction of aborted object. Type confusion and other implemented attacks can be prevented by: Runtime type checking, Object bound checking and Physical bound checking.

B. Countermeasures for Physical Attacks

Cryptographic countermeasures aim to get better implementation of cryptographic algorithms like RSA DES, and hash functions (MD5, SHA-1, etc.). Applicative countermeasures aim at the application code and generally produce application with bigger size to enabling embedding the security mechanism with the functional code of the application. E.g. pinverification. Systemcountermeasures targets to harden the system to offer a safe environment to the applications using method dual computation of sensitive data, use of execution counter, etc.

The various proposed methods for fault detection are:

- Field of Bit: This countermeasure considers that if an attack modifies an op-code by another one, it is possible that operands of the previous op-code are inconsistent with the new one.
- Basic Block: This protection method uses the basic block concept and the application control flow concept to check the code integrity.
- Path Checking: This detection method works on the byte code where all transformations and computations are done on a server (off-card).It checks the correctness of execution path.

C. Countermeasures for Combined Attacks

- Redundant checks: The most straightforward implementation of a fault detection mechanism on the stack would be to check the coherency between the value pushed onto the stack and the top of stack value after the push operation.
- Inducing errors**:** To ensure fault detection in approach to reduce the cost of redundant check is to propagate a potential error to another component of the JCVM

D. Countermeasures for other Attacks

- Use of multi layering to hide sensitive data lines
- Use of protective layeringto provide an active grid carrying protective signals to prevent the analysis of live data processing.
- Use of rigid sensors to disable the chips as soon as out of bound conditions are met.
- Bus Scrambling: using a sophisticated variant scrambling technique to prevent data line scrambling on cards.
- Use of balanced circuits and reduced EM signals to lower power signals for preventing side channel attacks
-

VII.EVALUATION AND ANALYSIS OF SMART CARD ATTACKS

Java Card versions 2.1.1, 2.2, or 2.2 was attacked to check the countermeasures circumvented [**9**] .Considering, the number of circumvented countermeasures a theoretic comparison on java cards has been made for logical attacks. The results given for combined attacks [**2**] form the basis for theoretic comparison. The fault attacks and their types have been compared on the basis of harm rendered to the card and cost of attack [**11**]. Combined attacks and their experimental results given in [**2, 1**] form the basis for theoretic comparison.

Table 1: Potency of Logical Sub attacks

Sr. No.	**Derived attack**	**Potency**	**Reference**
1.	Byte as short array	High	[9]
2.	Object as array	High	[9]
3.	Reference switching with AIDS	Medium	[9]
4.	Reference spoofing	Low	[9]

Table 1 presents potency of sub attacks which come under the category of logical attacks, first two attacks modify the byte and short with type confusion and third attack i.e. reference switching with AIDS results in complete applet duplication and may render the card useless.

Table 2: Potency of combined attack

Sr. No.	Combined Attack	Potency	Reference
1	Type Confusion	Low	[2]
2	Instance Confusion	High	[2]
3	Combined attack with ADPU tampering	High	[1]

Table2 presents potency of combined attacks on the basis of attack efficiency and state of card rendered. Type confusion and Instance confusion were carried out on card after conducting fault attacks on them. The results of these two have shown a phenomenal success. ADPU tampering leaves access privilegevulnerable and it becomes easy to access the card APDU buffer anytime, allowing reading its contents and modifyingthem.

Table 3: Potency of fault attacks

Sr. No.	Fault Attack	Potency	Ease of Attack	Cost of attack	Reference
1.	Spike Attack	High	Normal	Normal	[11]
2.	Glitch Attack	Medium	Normal	Low Cost	[11]
3.	Optical Attack	Medium	Hard	Low Cost	[7]
4.	Electromagnetic Attack	Medium	Hard	Low Cost	[11]

Table 3 presents potency of fault attacks as level of intrusion made to memory and amount of contents revealed. Fault attacks may lead to intrusion sometimes giving important information and may render card useless. The contents are exposed with scalping off the upper layer with some chemical solution i.e. removing upper layer to access the inner layer of card to expose to certain attacking modes. The cost of attack in terms of lab equipment comes out to be low and they are easy to carry out, but once again there useful only if some content is exposed and that is not the case every time.
Now based upon Table 1, Table 2 and Table 3 a comparative study of outlined java threats can be compared in table 4 on the basis of potency, average defense, cost and ease with which attack is conducted. Defense efficiency is considered on the basis of number of defense mechanisms, potency level is determined by number of attacks and sub attacks in a category their damage level, cost and ease of attack is evaluated on the basis of lab equipment and resources need to conduct an attack. The table above summarizes attacks on the basis of above said comparison measures. Table 4 gives the theoretic comparison with three major categories of attacks along with their sub attacks and the inferences made from their respective tables.

Table 4: Theoretic Comparison of attacks

Sr. No.	Attack	Sub Attack	Potency Level	Average Defense Efficiency	Overall Potency Level	Cost Of Attack	Ease of Attack
1.	**Logical Attacks**	CAP File Manipulation	Very High	Good	High	Normal	Medium
		Object Interface Abuse	Medium				
		Transaction Abuse	Low				
2.	**Physical Attacks**	Spike Attacks	High	Average	Medium	Low	Medium
		Glitch Attacks	Medium				
		Optical Attacks	Medium				
		Electromagnetic Attacks	Medium				
3.	**Combined Attacks**	Type Confusion	Low	Average	High	Normal	Hard
		Instance Confusion	High				
		APDU Tampering	High				

VIII.CONCLUSION AND FUTURE SCOPE

The study outlined the software architecture of a basic java card and highlighted the most potent category of attacks. Among all the attacks conducted logical attacks with CAP file manipulation bypasses major defensive techniques such as Physical Bound Checking, Runtime Type Checking, Object Bound Checking [**9**], and has also proved that code attacks are the most potent ones. Some of the techniques such as On Card BCV have also given better results but they need to be integrated into card during manufacturing process at some expense of money. Physical attacks are hard to conduct but can be conducted at low cost. Logical code attack and Fault attacks can be guarded against inherent mechanisms and can be used to derive out other new attacks. Hence, it can beconcluded that Logical code attacks are severe security threats to this technology of smart cards. For the future work Java Cards 3x architecture can be tested with Trojan applets with pluginsavailable at open sourceintegrated into Java tools like net beans or eclipse to identify the effectiveness of the latest security measures integrated into them to identify new security breaches.

REFERENCES

[1] Barbu, Guillaume, Christophe Giraud, and Vincent Guerin. "Embedded Eavesdropping on Java Card." *Information Security and Privacy Research*. Springer Berlin Heidelberg, 2012. 37-48.

[2] Barbu, Guillaume, Guillaume Duc, and Philippe Hoogvorst. "Java Card operand stack: fault attacks, combined attacks and countermeasures." *Smart Card Research and Advanced Applications*. Springer Berlin Heidelberg, 2011. 297-313.

[3] Chaumette, Serge, and Damien Sauveron. "An Efficient and Simple Way to Test the Security of Java CardsTM." *WOSIS*. 2005.

[4] Chen, Zhiqun. "Java card technology for smart cards: architecture and programmer's guide." *Addison –Wesley Professional*, 2000.

[5] Hogenboom, Jip, and WojciechMostowski. "Full memory read attack on a java card." *4th Benelux Workshop on Information and System Security Proceedings, WISSEC*. 2009.

[6] Hubbers, Engelbert, WojciechMostowski, and Erik Poll. "Tearing Java Cards." *Proceedings, e-Smart*. 2006.

[7] Quisquater, Jean-Jacques, and David Samyde. "Eddy current for magnetic analysis with active sensor." *Proceedings of Esmart*. Vol. 2002. 2002.

[8] Loizidis, Alexendros, Vasilios Almaliotis and Panagiotis Katsaros. "Static program analysis of multi-applet JavaCardApplication."*Department of Informatics.Aristotle University of Thessaloniki*, 2009.

[9] Mostowski, Wojciech, and Erik Poll. "Malicious code on Java Card smartcards: Attacks and countermeasures." *Smart Card Research and Advanced Applications*. Springer Berlin Heidelberg, 2008. 1-16.

[10] Sauveron, Damien. "Study and realization of an environment of experimentation and modelling for Java Card Technology ApplicationSecurity*." University Bordeux*, 2004.

[11] Sere, Ahmadou A., Julien Iguchi-Cartigny, and Jean-Louis Lanet. "Evaluation of Countermeasures against Fault Attacks on Smart Cards." *International Journal of Security & Its Applications* 5.2 (2011).

[12] Sere,Ahmadou Al Khary, Julien Iguchi- Cartigny , and Jean – Louis Lanet. "Automatic detection of fault attack and countermeasures." "*Proceedings of the 4th Workshop on Embedded System Security*. ACM, 2009.

[13] Witterman, Marc. "Java Card Security." *Information Security Bulletin*, 2003.

WEB REFERENCE

[14] http:// java.sun.com/products/java card2.2.2/specs.html(accessed on 02.08.2014).

