

# A Comparison of the Performance and Security of Free and Open Source Smart Card Offcard APIs

Albert Chifura\*  
[achifura@ufh.ac.za](mailto:achifura@ufh.ac.za)

Peter Nkomo  
[pnkomo@csir.co.za](mailto:pnkomo@csir.co.za)

Jim Chadwick  
[jchadwick@ufh.ac.za](mailto:jchadwick@ufh.ac.za)

Department of Computer Science, University of Fort Hare, Alice, 5700

\*Primary Author for correspondence: TEL/FAX 0406022464

Conference Topic: Network Services    Sub Topics: E-Commerce, Web services

## Abstract

**The increase in internet connectivity has created new avenues for smart cards to be used as identity tokens in distributed on-line web-services. To make this possible, one must be able to read the card and communicate with it using an application. Off card APIs provide a framework for this by providing interfaces that can be implemented to enhance the communication. This study looks at the performance and security comparisons of several free and open source smart card offcard APIs that are currently in existence.**

*Keywords:* Smart card, open source offcard APIs, performance evaluation, security evaluation

## 1. INTRODUCTION

Increased internet connectivity has promoted a migration from paper based and face -to-face service delivery to the use of Internet in order to provide customers with the most convenient way of accessing goods and services [1]. This migration has necessitated the exploration of new forms of secure on-line identification. The adoption of the Internet has raised the question of on-line security because the Internet is insecure. Internet security has increased the need for a centralized access control and strong authentication methods. Distributed web-services and the need for both a centralized administration and a single sign-on solution have become even more relevant to relieve the user of the need to remember multiple user credentials. Smart cards have been identified as suitable forms of identity tokens to use in this scenario.

## 2. BACKGROUND

A smart card is a portable, tamper-resistant computer with a programmable data store. It is the exact shape and size of a

credit card, but can hold 4KB - 64KB of information and perform a modest amount of data processing as well [2]. Smart cards offer a lot of advantages. They are portable, resistant to attack, easy to use and allow secure storage of security keys for encryption and decryption. On distributed on-line services, smart cards solve the problem of having to manage and track authorized users by providing a single-sign on.

For smart cards to be used in a PC environment in distributed web services, it is necessary to have a terminal that is connected to the PC and to have support from the PC software. A smart card can be seen as a small computer whose role is to respond to commands from the PC, also called the host. The application that is used to communicate with the card resides on the PC or terminal and is therefore outside the card. Development of such applications has led to the creation of various off-card application programming interface (API). These API provide an interface through which the card and the PC application communicate.

While the adoption of smart cards provides a solution to single-sign on and security on-line, it also presents a performance challenge to developers. Smart cards have low processing power and limited memory. As a result developers have to take into consideration factors that add substantial delays to the execution of their applications. These delays can be encountered both on the smart card itself and when communicating with the card. Research has concentrated very much on improving on-card execution as documented in [3]. As [4] has pointed out, delays on the card can occur, for example, when the application frequently writes to EEPROM instead of RAM during execution, since this is slower [4]. Significant delays can also result from cryptographic algorithms. The latter has been the area that has attracted much research in an effort to improve execution time of smart cards because of their low computing power.

While there is a need to improve the performance of the cryptographic functions and design programs that consume less memory, there is also a need to consider factors outside the smart card itself. Smart cards work in a slave-master half-duplexed communication with the smart card playing the passive or slave role waiting for commands from the host. This is so because the smart card microprocessors have a single input/output port. This means that smart card application delays do result from sending and receiving data packets from the host to the card and from the card to the host. The emergence of technologies to enhance communication between smart cards and card acceptance devices (CAD) raises an area worth exploring.

### 3. AIM OF THE RESEARCH

Smart card Off-card APIs have become more complex in architecture in an effort to hide details of the underlying operating system of the terminal [3]. This has the potential to add communication overhead. It follows that there is a need to investigate whether there are any delays introduced by the off-card APIs during communication between the host application and the smart card. The areas worth exploring are:

- Whether the off-card API does indeed add delays in the communication between the smart card application and the application on the card acceptance device.
- Whether one particular API is superior to the other in terms of speed, general performance and security.
- Whether there is more effort required to improve the smart card communication API, as it is mainly used during the execution of the smart card application.

### 4. IMPLEMENTATION TOOLS

This research is mainly based on laboratory measurements and data transfer tests. To carry out these measurements and tests the research involves setting up an experimental environment that will implement PC/SC [5], OCF [6], Global Platform [7], CT-API [8] and Native drivers [8]. These will be installed on a windows 200 Professional machine with 512 RAM, 40GB hard disk and 1.80 GHz of CPU. The following hardware and software will also be used:

- JCOP tools
- Eclipse
- Precise Biometric 100MC smart card terminal

### 5. INTENDED OUTCOME OF THE RESEARCH

At the end of the project a testing environment should be set up that will implement PC/SC, OCF, Global Platform, CT-API and Native Drivers to enable these technologies to be tested. Advice on which API is suitable for a particular application will be given and also the obtained results should help in giving advice on how different types of APIs can be combined to give the best possible performance or security when designing web based smart card applications.

### References

- [1] Eric Rocco, "The Internet and Service Delivery: Great Expectations", available from. <http://www.culpepper.com/eBulletin/1996/153rocco.asp>
- [2] Nicole Shillington & Travers Waker, "The design of a smart card interface device", available from. <http://www.cs.uct.ac.za/Research/DNA/SOCS/rchap2.html>
- [3] Constantious Markantonakis, "Is the performance of smart card cryptographic functions the real bottleneck", Proceedings of the 16th international conference on Information security, p.77-91, September, 2001
- [4] Zhiquan Chen, "Java Card Technology for Smart Cards", Addison-Wesley, 2000
- [5] PC/SC Workgroup, "Specifications for PC-ICC Interoperability", Available from. <http://www.pcscworkgroup.com/>
- [6] Uwe Hansmann, Martin, S. Nicklous, Thomas Schack, Achim Schneider, Frank Seliger, "Smart Card Application Development Using Java", Springer, 2002.
- [7] Global platform, "Introduction to global platform standards", Available from. <http://www.globalplatform.org/>
- [8] Vesna Hassler, Martin Manninger, Mikhail Gordeev, Christorh Muller, "Java Card for e-payment Applications", Artech House, 2002

### Biographical Note



**Albert Chifura** received his BSc Honours degree (*Cum laude*) in Computer Science from the University of fort Hare. He is presently doing his MSc in Computer Science at University of fort Hare