Secure Element Access from a Web browser

Position paper for W3C Workshop on Authentication, Hardware Tokens and Beyond, September 10th-11th 2014.
Bruno JAVARY b.javary@oberthur.com, R&D, Identity BU, Oberthur Technologies
Florian RECAPE, R&D, Identity BU, Oberthur Technologies

EXECUTIVE SUMMARY

In order to use all secure elements features, some technologies to interact with the Smart Card in different form factors have been developed (middleware, plug-ins, specific applications).
Considering development of web applications and mobility, we can assert that standard and customized solutions are acceptable for the PC in a controlled environment but are disputed by the revolution of Smartphone and tablets usage.

To enable a common access for every single user to trusted services thanks to a secure element, the best candidate is the web browser. The web browser and by consequence HTML and JavaScript will be the standard to access a secure element as it was done already with webcam, video, gps, … They need to evolve in a more secure and convenient ways.

INTRODUCTION

In 2013 about 7 billion of Smart Cards have been sold around the world (source: Eurosmart). Even if Telecoms represent 68% of the market, the market is moving and we can find Smart Card everywhere now. Smart cards and more generally secure tokens are now able to embed a lot of cryptographic functionalities and personal information. In order to bring convenience and security to users in a daily life, some technologies to interact with the Smart Card have been developed.

For the PC there are a lot of solutions depending on whether we want to use the Smart Card online or on a local use. If we want to use the secure element online, most of the solutions are web browser extensions, or Java Applet. These solutions extend the browser features by giving ability to communicate with hardware components like a webcam, file system... On a local use, it's better to use a middleware which is responsible for managing the connection between the secure element and the software running on computer which needs security.

On mobile (Smartphone or tablet), the situation is different because we don't have any standard way to interact with a secure element, there is no real Middleware and web browser extensions are not compatible.

10 years ago when we wanted to watch a video from a website, a Flash plug-in was required, now service providers and browsers editors defined a standard to make the browser self sufficient and use it since. We don't teach anyone with this but this is important to point out that hardware can be accessed through Web browser natively.

Today, thanks among other to the work done by System applications Working Group, many hardware are available: webcam, gps for geolocation, …

EXISTING : WHAT ARE THE DRAWBACKS

A middleware is a software application that enhances the capacities of our computer applications by creating an abstraction layer. This abstraction layer will help the PC software communicating with a secure token focusing on its cryptographic capabilities. Thanks to a middleware it is possible to log on to a computer with a secure token, trigger actions when it is removed, sign or decrypt a document…

A middleware implements standards as PKCS#11 or MiniDriver to communicate with a secure element. These standards provide to all developers a common pipe of access to a Smart Card or alike, which increases the security by reducing the heterogeneity of access methods. All applications needing trusted services share a common high level of security defined by trusted authorities.

Nevertheless, it's only possible to use a middleware in a limited scope and PKCS#11 does not fit with distributed architecture. For example if a user wants to use his secure element to decrypt a document online, he must first download it in order to proceed to the decryption in local. In addition a middleware is a full software which requires an installation.

Middleware is a good solution for a local use, it provides secure features established on standards in a controlled IT configuration. However it can't be used as an online solution or in an opened device.

Another method to access Smart Card from a computer is to use a web browser extension. A web browser extension is a program integrated into a web browser and which provides new features. Especially they allow browser to access hardware devices (file system, webcam…).

In order to access a Smart Card from a web browser there are currently two different kinds of solutions, the implementation of a plug-in on a browser, or the use of a Java Applet. The two solutions are quite similar excepted that a Java Applet is independent of the browser used as it's running in a JVM while a plug-in must be adapted.

Web browser extensions are currently the only way to access a secure token online; even so they present many drawbacks that make them unsuitable for a public use. Their main problem is that they are not subject to any standard so there is a large heterogeneity of methods to access Smart Card as every developer can make his own. The variety of

solutions and uncontrolled access to user's workspace limits security and trust level of overall solution. Each service provider will provide its own plug-in with its proprietary access methods, these actors could be more or less trusted. Running of a plug-in is actually controlled but access rights are not differentiated by field of operation, the user authorizes the plug-in to run without ability to control the scope of its execution. This is the reason why these extensions are considered as a threat for the security in browsers. At last, most of them are not available on mobile devices.

On mobile devices, the situation is different because accessing a secure element is possible only with proprietary apis or NFC and is limited to low level communication. Exception has to be made for Open Mobile API which enables mobile applications to have this kind of access but implementations are quite limited for the moment. Unlike PC environments, existing Middleware and web browser extensions do not fit in a mobile environment. This makes mobile applications very specific in the sense that they are today kindly not reusable and implements targeted use cases.

USE CASE : PIV

Let's take the example of a US federal employee or contractor owning a PIV card defined by the National Institute of Standards and Technology (NIST). The card is required to enter a governmental building and to log on to computers (Physical and Logical Access Control). The federal employee can also sign emails or documents and authenticates to remote web sites in HTTPS.

The card is used by dedicated software using directly "smart card language" (APDUs) to communicate with it or by standard software using a set of common functions (middleware mechanism).

Nevertheless this model is facing several limitations that impact its usability:

- File decryption or signing must be done locally. In a world of cloud computing and "Software as a Service" it represents a real inconvenience.
- The agent must have an already configured PC or be granted with specific rights, which prevents from using devices "on the go" or "away from office" (in a hotel, an airport, at home).
- To use a Smartphone or a tablet, specific software and hardware (card reader) have to be set up.

With a standard access to a secure element from a web browser, this user could access the same web page from his different devices (PC, mobile, tablet) natively, with the same high level of security, and have a more convenient experience.

This use case is very specific but many applications could be found, involving a wider share of population like mainstream market products, for example:

- Ticketing, Couponing: access to concert / movie thanks to smartphone, loyalty cards
- eBanking: Online authentication, enrolment, "Know Your Customer"
- eGovernment: Tax declaration, Citizen e-services.

PERSPECTIVE AND PROPOSAL

Standard and customized solutions are acceptable for the PC in a controlled environment. Each of these solutions has advantages and drawbacks but this figure is completely questioned by the Smartphone and tablets market share evolution.

This mobility revolution strongly needs standardization to offer security. What has been done with PKCS#11 and Minidriver must be transposed to mobile operating systems.

OT's position is to be a leader of digital security solutions for the mobility space. This means that OT is providing solutions with secure elements (such as smartcard, UICC, microSD, embedded secure element) and is promoting these solutions to be used by many services in a wide range of devices. In a mobile environment, using NFC's smart card reader mode is promising, hardware and software are available, access rights and communication are managed in a standard way.

As a solution provider, we would like to push the standardisation of a JavaScript API which allows web browser to communicate with Smart Card in order to use it in OT solutions and also to encourage partners to deploy secured and trusted solutions combined with standard developments. This way, OT is willing to open trusted services with secure element to the mainstream market. In order to be implemented in all browsers and to ensure its liability, the API should be endorsed by W3C.

Opening the browser to secure element allows a unique layer of access available to the widest range of developers with technology allowing rich content and widely deployed and known: HTML5. Web actors as service providers could offer security services with any kind of token, increasing user experience, convenience and security for the consumer.

This way, the sensitive security layer will be managed by trusted stakeholders who are browser editors and will be

used by web developers. The website on its side must have a limited, controlled and acknowledged access to cryptographic features of a secure token.

As HTML5 is cross-platform the api will work on every environments and devices, whether on a PC, a Tablet, a Smartphone, or even a Smart TV (who knows?).

A proposal has already been drafted regarding secure element access: Secure Element API (http://opoto.github.io/secure-element/).

This api is complete and well documented. It presents in details the technical background and use cases and gives a good visibility of Security, Permissions, Access Control and Conformance. OT will follow and apply it in future developments.

Security is at the heart of OT's concerns; the proposed solution combines validation of the feature by the user and a specific access control mechanism. It allows the web site and the application in the smart card to validate that the communication is genuine by implementing both the runtime and security model for web applications and Access Control Enforcer defined by Global Platform. The idea beyond is to propose a trusted access to a secure element from a service provider, preventing from unauthorised use.

Let's follow, jointly with all companies sharing the same opinion and interest, action plan below:

- Identify a charter to carry the project
- Define use cases and for each of them demonstrate the impact and validate the consistency of the current proposal.
- Meeting all stakeholders interested in the subject, be aware of each of them interest and create a common basis of communication and strategy
- Establish interactions with other standardisations (eg Open Mobile API)
- Gather work forces to create a proof of concept and decline it to use cases examples (eg eServices)

BIBLIOGRAPHY

[1] Eurosmart reference *http://www.eurosmart.com/publications/market-overview*
[2] W3C Sysapp working group *http://www.w3.org/2012/sysapps/*
[3] Minidriver *http://msdn.microsoft.com/en-us/library/windows/hardware/dn468773(v=vs.85).aspx*
[4] PKCS#11 *http://www.emc.com/emc-plus/rsa-labs/standards-initiatives/pkcs-11-cryptographic-token-interface-standard.htm*
[5] Middleware implementation example *https://github.com/OpenSC/OpenSC/wiki*
[6] Browser security Wikipedia page *http://en.wikipedia.org/wiki/Browser_security*
[7] An example of security requirement *http://www.us-cert.gov/publications/securing-your-web-browser*
[8] Oberthur web site *http://www.oberthur.com/*
[9] Oberthur corporate solution *http://www.oberthur.com/presse/ot-launches-quardialys-convergence-its-new-secure-access-control-solution-through-a-unique-secure-device/*
[10] Oberthur PIV *http://www.oberthur.com/presse/oberthur-technologies-offers-a-user-authentication-solution-to-governments-and-corporations-leveraging-on-its-embedded-secure-element-ese-in-mobile-devices/*
[11] PIV standard *http://www.nist.gov/itl/csd/ssa/piv.cfm*
[12] Know Your Customer definition http://en.wikipedia.org/wiki/Know_your_customer
[13] Open Mobile API *http://www.simalliance.org/en/handset/*
[14] Seek for android (implementation of Open Mobile API) *https://code.google.com/p/seek-for-android/*
[15] Secure Element Access Control Enforcer *https://code.google.com/p/seek-for-android/wiki/AccessControlIntroduction*
[16] Secure Element API http://opoto.github.io/secure-element/