Firmador digital optimizado para aplicaciones web

Diego Crivelli, Leandro Quiroga

{dcrivelli[1 2], lequir[1 2]}@mecon.gov.ar

[1] Dirección General de Sistemas Informáticos de Administración Financiera (DGSIAF)
Subsecretaría de Presupuesto, Secretaría de Hacienda, Ministerio de Hacienda y Finanzas
Balcarce 186, CABA, Argentina

[2] Laboratorio de Investigación y Formación en Informática Avanzada (LIFIA)
Facultad de Informática, Universidad Nacional de La Plata (UNLP)
Calle 50 y 120, La Plata, Buenos Aires, Argentina
http://lifia.info.unlp.edu.ar

Resumen. El objetivo principal de este trabajo fue diseñar e implementar un firmador web que posibilite el desarrollo de aplicaciones web de Administración financiera, orientadas a autoridades de los Organismos Nacionales que acceden al eSidif (Sistema integrado de Administración financiera) casi exclusivamente para firmar documentos digitalmente de forma masiva. Fueron varios los desafíos que se presentaron al emprender este proyecto. Entre los más importantes se destacan: la necesidad de firmar documentos sin transferirlos al cliente para evitar problemas de *performance*, la utilización de dispositivos criptográficos homologados y la obtención de una solución ágil, segura y simple de acceder.

Palabras clave: Firma digital, *hash*, cliente web, dispositivo criptográfico, PDF

1 Introducción

El eSidif es un sistema de gran tamaño, diseñado para sustentar la diversidad de tareas y roles que requiere la administración financiera de la Administración Pública Nacional. Presenta una arquitectura en capas: clientes, servidor y datos. El servidor sigue un diseño orientado a servicios sobre tecnología Java. La capa de datos se implementa sobre una base de datos Oracle. Entre los clientes del servidor eSidif se destacan sistemas que consumen *web services* y el cliente de escritorio que utilizan a diario los usuarios operativos de este sistema. Este cliente, está basado en tecnología Java-RCP (Rich Client Platform). Para evitar la instalación de este cliente en las computadoras de los usuarios y hacerlo accesible a través de la web, se utiliza una tecnología de virtualización (Citrix).

Dado que la legislación le otorga a una firma digital el mismo valor legal que a una firma hológrafa, desde el año 2011 la Subsecretaría de Presupuesto de la Secretaría de Hacienda impulsa—a través del eSidif—la *despapelización* de diferentes procesos

relacionados con la Administración Financiera. Esta *despapelización* favorece el reemplazo de expedientes en papel y agiliza las gestiones administrativas.

En la versión original del firmador, para firmar digitalmente un documento, los usuarios acceden a un cliente Citrix y luego se autentican en el eSidif. En el contexto de una gestión que requiere firma digital, el sistema genera en el servidor el documento PDF a firmar y lo retorna al cliente. Para aplicar la operación de firma, el usuario introduce el dispositivo criptográfico (*token*) que contiene el certificado digital, propiedades del usuario, y firma el documento. Luego, el cliente envía el documento firmado al servidor para su persistencia. Dado que tanto el cliente como el servidor se ejecutan en servidores internos y con una buena infraestructura de red, estas transferencias de documentos no presentan inconvenientes en cuanto al tiempo de respuesta de la operación.

Existen determinados usuarios, en su mayoría autoridades, que casi con exclusividad utilizan el eSidif para firmar comprobantes digitalmente. Para este tipo de usuarios, se pretende ofrecer una solución (a partir de aquí *Bandeja de firmas web*) de acceso simple, orientada a resolver exclusivamente las necesidades de firma, simplificando de esta forma su operatoria.

Para la implementación de la *Bandeja de firmas web*, fue imprescindible desarrollar un firmador web, el cual es el objeto de este trabajo.

2 Solución

2.1 Arquitectura

Se desarrolló una aplicación cliente-servidor. El servidor consiste en un *servlet* implementado en Java y desplegado en un servidor Jboss. Se utiliza Hibernate y una base de datos Oracle para la persistencia. Para la manipulación de los archivos PDF, se utilizó la librería iText (http://itextpdf.com/), mientras que para los procesos criptográficos se utilizó la librería Bouncy Castle (https://www.bouncycastle.org/).

Para implementar el cliente, se optó por un *applet* Java. Aunque ésta es una tecnología en desuso, fue la única que nos brindó la seguridad necesaria para acceder e interactuar desde un navegador web con dispositivos USB como los *tokens*. En el futuro, se intentará reemplazar dicho *applet* con el uso de tecnologías JavaScript, como Node.js (https://nodejs.org/) y socket.io (http://socket.io/), librerías que actualmente están siendo utilizadas para la comunicación con distintos tipos de dispositivos.

2.2 Funcionamiento

Al entrar a la bandeja de firmas web se le presentan al usuario todos los comprobantes que tiene a su firma. El usuario selecciona los que desee firmar y oprime el botón *Firmar* para llevar a cabo la operación. La aplicación cliente envía al servidor los identificadores de los comprobantes a firmar. Una vez recibidos los identificadores, el

servidor obtiene los comprobantes, los transforma en documentos PDF, calcula un *hash* criptográfico para cada documento y lo devuelve al cliente. Este *hash* es una representación liviana del documento a firmar que lo identifica unívocamente. Se utiliza la librería iText tanto para la generación del documento PDF a partir del comprobante como para la obtención del *hash.*

Los *hashes* firmados constituyen la firma digital de los documentos PDF. Solo resta adjuntar estas firmas digitales a los documentos a los que pertenecen. Así, el cliente invoca al servidor nuevamente y le envía los *hashes* firmados. El servidor adjunta los *hashes* a los documentos PDF y luego los persiste en la base de datos.

Gracias a esta solución, la interacción con el dispositivo criptográfico queda ligada solo a la aplicación cliente, mientras que el servidor es el único que interactúa con los documentos PDF. Cabe aclarar que si el usuario quisiera revisar los documentos PDF generados, estos indefectiblemente deberán transferirse al cliente web.

2.3 Descripción técnica

Para crear el firmador, primero el cliente debe establecer comunicación con el dispositivo criptográfico USB (*token*) utilizando el estándar PKCS#11. Si trabajamos con Windows, necesitaremos la ruta al archivo DLL correspondiente al *driver* del dispositivo; en el caso de Linux, necesitaremos la ruta al archivo.so. Luego, procedemos a escanear los puertos en busca del dispositivo criptográfico conectado. Una vez que hemos establecido la comunicación con el dispositivo, seleccionaremos el certificado o, si hubiere más de uno, lo seleccionará el usuario según su alias.

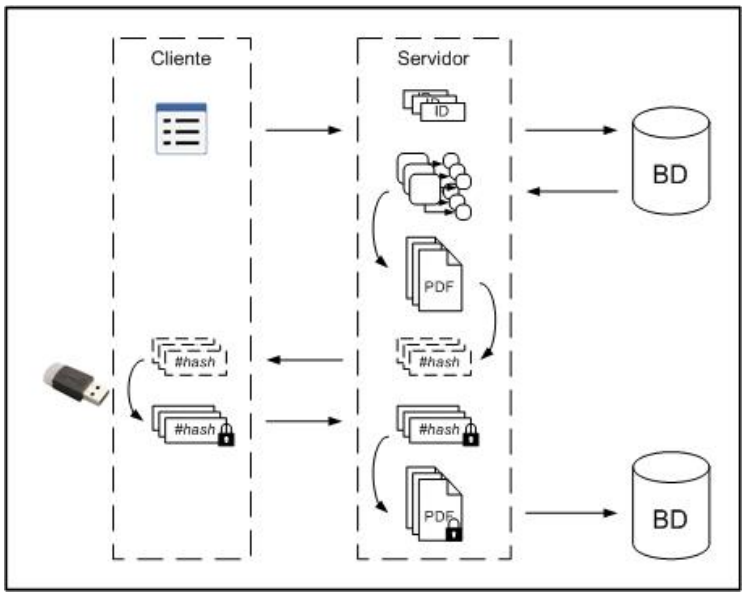

Fig. 1. Diagrama de secuencia del firmador web.

A continuación, el cliente le solicita al servidor el *hash*, o los *hashes*, del PDF a firmar (generado con el algoritmo SHA-1) y le envía el certificado antes obtenido. El servidor recibe la solicitud y entonces prepara un nuevo documento PDF que será el que finalmente se firme. Para ello, se genera un *reader*, un *stamper* y un *appearance* y se procede a crear la infraestructura que tendrá la firma. Todos estos datos son almacenamos en memoria, ya que luego serán utilizados para generar el documento resultante. Al finalizar este proceso, el servidor responde al cliente con los *bytes* del *hash* que necesitan ser firmados; el cliente los recibe e interactúa sucesivamente con el *token* USB para firmarlos utilizando la clave privada que lee desde un *KeyStore*. Es importante destacar que en el cliente no se utiliza la librería iText (pero sí en el servidor) para firmar estos *bytes*, sino la librería interna, *java.security.Signature*. De regreso en el servidor, los *bytes* firmados son recibidos: se completa el proceso de firma del archivo PDF con la infraestructura antes generada, los *bytes* y la clave pública del certificado. Por último, se persiste el nuevo archivo PDF en la base de datos.

La aplicación fue probada con los dispositivos criptográficos eToken pro 5100 y MS-IDProtect token USB. Ambos cumplen con la certificación FIPS 140-2 tal como lo exige la normativa vigente de firma digital regulada por la ONTI (Oficina Nacional de Tecnologías de Información).

3 Conclusión

La *Bandeja de firmas web* brinda la posibilidad de firmar documentos digitalmente otorgando la misma seguridad que la aplicación de gestión del presupuesto nacional (eSidif). Además, la separación del proceso de firma en el cliente y el servidor permitió reducir drásticamente la transferencia de información entre ambos. Es evidente el potencial de esta herramienta—pronto podrá estar a disposición de cualquier aplicación que necesite integrar un firmador de estas características.

En un futuro, esperamos poder cubrir la necesidad de firmar documentos con dispositivos móviles, ya que la solución puede adaptarse fácilmente a otras tecnologías, como los certificados alojados en servidores (HSM), que en Argentina aún no han sido homologadas.

Referencias

1. Acrobat Family of Products: Digital Signatures in a PDF, http://www.adobe.com/devnet-docs/acrobatetk/tools/DigSig/Acrobat_DigitalSignatures_in_PDF.pdf
2. Lowagie B.: Digital Signature for PDF Documents 52-64, 94-123 (2012).
3. Lowagie B.: iText in Action. Manning Publications Co (2011).