La seguridad en el Servicio de firma del Cliente @firma mediante invocación por protocolo

Versión 1.4

Tabla de contenido

[Introducción 3](#_Toc433741884)

[Aplicación AutoFirma 3](#_Toc433741885)

[El siguiente diagrama ilustra un flujo de operación de firma básico: 3](#_Toc433741886)

[Invocación por protocolo 3](#_Toc433741887)

[¿Qué es la invocación por protocolo? 3](#_Toc433741888)

[Invocación por protocolo desde navegador Web 4](#_Toc433741889)

[Comunicación entre aplicación nativa y aplicación JavaScript en el navegador que inició la invocación por protocolo 6](#_Toc433741890)

[Funcionamiento del diálogo entre aplicación AutoFirma y aplicación JavaScript en navegador Web 6](#_Toc433741891)

[Aplicación AutoFirma con interfaz gráfico para firmas locales 7](#_Toc433741892)

[Seguridad de la aplicación AutoFirma 7](#_Toc433741893)

[Seguridad en el propio instalador 7](#_Toc433741894)

[Microsoft Windows 7](#_Toc433741895)

[Apple OS X 7](#_Toc433741896)

[Linux 7](#_Toc433741897)

[Cambios realizados en el sistema durante el proceso de instalación 7](#_Toc433741898)

[Registro del protocolo “afirma” 7](#_Toc433741899)

[Instalación de un certificado SSL auto firmado como de confianza para el sistema 8](#_Toc433741900)

[Otros cambios 8](#_Toc433741901)

[Seguridad en el propio binario ejecutable de AutoFirma 9](#_Toc433741902)

[Seguridad en el propio proceso de firma 9](#_Toc433741903)

# Introducción

## Aplicación AutoFirma

La aplicación AutoFirma es un programa que permite la realización de firmas electrónicas integradas en flujos de trabajo Web accesibles por parte de los usuarios firmantes mediante un simple navegador Web.

Las firmas electrónicas se realizan siempre en el equipo local del firmante, accediendo de forma segura a las claves privadas y certificados de firma almacenados en sus repositorios locales (del navegador, del sistema operativo, en tarjetas inteligentes, etc.).

Al contrario que las aplicaciones comunes de firma electrónica Web, AutoFirma no necesita que el navegador Web soporte Applets de Java, si bien es necesario que esta aplicación esté instalada en el ordenador local del usuario firmante antes de iniciar el trámite de firma electrónica.

Para suplir la necesidad de soporte de Applets de Java, la aplicación AutoFirma y la lógica JavaScript del flujo Web del trámite de firma electrónica integran un JavaScript distribuido junto AutoFirma (miniapplet.js). Mediante este JavaScript se ejecutan las distintas operaciones de firma, para lo cual se arrancará la aplicación usando un mecanismo denominado Invocación por Protocolo. Una vez arrancada la aplicación se iniciará un diálogo bidireccional a través de un socket entre el JavaScript de despliegue y AutoFirma. Por medio de este socket se trasladarán a AutoFirma la orden de firma correspondiente y los datos sobre los que operar.

## El siguiente diagrama ilustra un flujo de operación de firma básico:

Firma

Página Web

miniapplet.js

AutoFirma

firmar(datos)

Firma

firmar(datos)

invocarPorProtocolo(id, puertos)

## Invocación por protocolo

### ¿Qué es la invocación por protocolo?

Es un funcionamiento universal que los sistemas operativos mantengan una serie de asociaciones entre tipos de fichero y las aplicaciones que son capaces de tratarlos. Así, si en un sistema operativo Windows se indica que se abra un documento de texto, este consultará en el Registro de Windows cual es la aplicación por defecto asociada para su tratamiento (usualmente el Bloc de Notas), y procederá a abrir esta aplicación pasando como parámetro la ruta completa del fichero en el esquema de argumentos definido en el propio Registro de Windows como parte de la asociación.

Esta asociación se hace de distintas formas según el sistema operativo, en Windows es por su extensión (“.txt” en nuestro ejemplo), pero por ejemplo en Linux es por su MIME-Type (text/plain en el ejemplo).

Este mismo esquema se define igualmente en la mayoría de los sistemas operativos también para los esquemas comunes de protocolos basados en URN/URI/URL. Así, si por ejemplo en un sistema operativo Windows indicamos que queremos abrir http://www.google.com (por ejemplo, desde la línea de comandos con la sentencia “start http://www.google.com”) se iniciará el navegador Web por defecto, que es la aplicación asociada para tratar el protocolo http, procediendo a abrir esa página Web.

Este modo de abrir aplicaciones se conoce como invocación por protocolo, y de forma análoga a la invocación de aplicaciones indicando abrir un fichero, donde antes se recibía la ruta completa del fichero a abrir, ahora se recibe la URL/URI/URN completa que se indicó abrir.

La invocación por protocolo permite enviar parámetros a la aplicación de destino como parte de la URL de invocación. Sin embargo, no es posible transmitir datos desde la aplicación invocada a la aplicación que la llamó, por lo que no puede devolverse ningún resultado.

### Invocación por protocolo desde navegador Web

Este mecanismo de invocación por protocolo de los sistemas operativos es usualmente accesible desde los navegadores Web. Esto quiere decir que si en la barra de direcciones del navegador Web indicamos una URI, el navegador Web trasladará el control al sistema operativo para que este localice la aplicación apropiada para tratar el protocolo asociado a la URI, y la abra pasándole dicha URI.

Un ejemplo de este mecanismo en Apple iOS podría ser el soporte del protocolo tel en forma de URN con el formato tel://1-408-555-5555, donde 1-408-555-5555 es un número de teléfono. Así una llamada desde una página Web a esta URN con una sentencia HTML como la siguiente, <a href="tel://1-408-555-5555">1-408-555-5555</a>,provoca que se active el teléfono (en un iPhone) y realice una llamada a ese número, ya que la aplicación nativa de teléfono de iOS tiene registrado ese esquema de protocolo.

En este caso tenemos una salvedad evidente, y es que el navegador Web obviará esta transferencia de control al sistema operativo cuando el propio navegador sepa tratar el protocolo, por ejemplo, con http, https, ftp, etc.

#### Advertencias de apertura

Como la invocación por protocolo no deja de ser una transferencia de datos desde una página Web (que no tiene porqué ser de confianza) a una aplicación nativa, los navegadores Web acostumbran a advertir de este cambio al usuario con un diálogo gráfico:

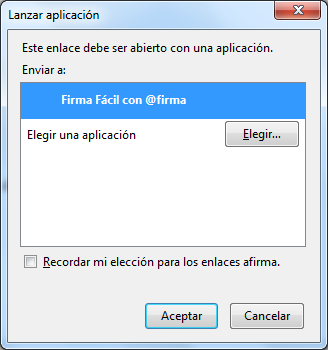


Ilustración 1: Advertencia en Firefox sobre Windows 7

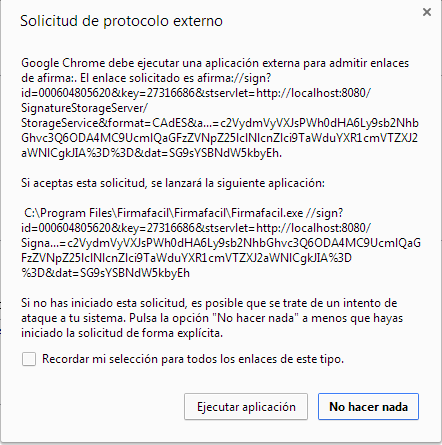


Ilustración 2: Advertencia en Google Chrome sobre Windows 7

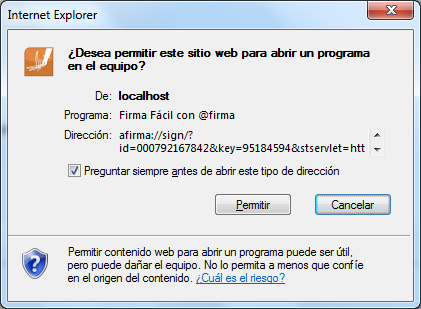


Ilustración 3: Advertencia en Internet Explorer 9 sobre Windows 7

En general, todos los navegadores Web muestran algún tipo de advertencia, excepto Apple Safari en Windows y OS X. En el caso de Internet Explorer se comprueba además la firma electrónica del ejecutable del programa nativo invocado.

#### Soporte de la invocación por protocolo en distintos navegadores Web

Las pruebas de compatibilidad positiva se han realizado en los siguientes entornos de escritorio compatibles con AutoFirma:

* Windows (XP, Vista, 7, 8, 8.1)
* OS X (10.7, 10.8, 10.9)
* Linux (varias distribuciones y versiones).

### Comunicación entre aplicación nativa y aplicación JavaScript en el navegador que inició la invocación por protocolo

Como hemos comentado, una aplicación JavaScript ejecutándose en un navegador Web puede invocar una aplicación nativa siempre que esta esté registrada como la aplicación por defecto para tratar un protocolo que no trate el propio navegador (por ejemplo, el Cliente @firma usa el protocolo afirma en una URI del estilo afirma://), proporcionando ciertos datos como parte de la propia URI de invocación, pero no hay mecanismos predefinidos para, tras esta invocación, establecer un diálogo bidireccional.

## Funcionamiento del diálogo entre aplicación AutoFirma y aplicación JavaScript en navegador Web

Dado que, como se ha comentado anteriormente, no hay un mecanismo predefinido para la comunicación entre una aplicación nativa ajena al navegador Web y este último, en AutoFirma se ha implementado un procedimiento de comunicación mediante HTTPS, donde la aplicación AutoFirma actúa como servidor (mediante un *ServerSocket* SSL de Java) y la aplicación JavaScript realiza llamadas HTTPS normales, dirigidas siempre hacia la dirección 127.0.0.1, que siempre corresponde a la máquina local del firmante.

El puerto del *socket* a través del cual se realiza la comunicación se genera aleatoriamente en el JavaScript de integración y se envía a AutoFirma mediante una invocación por protocolo. A partir de este momento se inicia un diálogo bidireccional a través del *socket* SSL.

### Aplicación AutoFirma con interfaz gráfico para firmas locales

La aplicación AutoFirma permite su ejecución directa en local por parte del usuario para la generación de firmas electrónicas sin necesidad de su integración en un flujo Web. Sin embargo, este modo de ejecución está fuera del alcance de este documento.

# Seguridad de la aplicación AutoFirma

## Seguridad en el propio instalador

### Microsoft Windows

El instalador (.exe) va firmado electrónicamente, y MS-Windows es capaz de comprobar la validez de la firma.

### Apple OS X

El instalador (.pkg) va firmado electrónicamente según los requisitos de Apple para la generación de instaladores firmados.

### Linux

No hay firma electrónica, al no soportarse esta en los instaladores DEB de Debian.

## Cambios realizados en el sistema durante el proceso de instalación

### Registro del protocolo “afirma”

Una de las tareas realizadas por el instalador es el registro del protocolo “afirma” asociado a la aplicación AutoFirma, de modo que una invocación al sistema operativo para la apertura de una URL del tipo “afirma://” conllevará el arranque de la aplicación AutoFirma, recibiendo esta la URL completa con la que fue instanciada.

Este registro se realiza de distinta manera según el sistema operativo:

#### Microsoft Windows

El registro del protocolo se realiza mediante el propio programa de instalación (NSIS, Nullsoft Scriptable Install System), que ejecutándose con privilegios de administrador, añade los siguientes valores en el registro de Windows (sintaxis según NSIS, donde $INSTDIR es el directorio de instalación de AutoFirma):

;Protocolo afirma

WriteRegStr HKEY\_CLASSES\_ROOT "afirma" "" "URL:Afirma Protocol"

WriteRegStr HKEY\_CLASSES\_ROOT "afirma\DefaultIcon" "" "$INSTDIR\AutoFirma\ic\_firmar.ico"

WriteRegStr HKEY\_CLASSES\_ROOT "afirma" "URL Protocol" ""

WriteRegStr HKEY\_CLASSES\_ROOT "afirma\shell\open\command" "" "$INSTDIR\AutoFirma\AutoFirma.exe %1"

Estos valores son eliminados durante la desinstalación, igualmente por el desinstalador de NSIS ejecutándose con permisos de administrador.

#### Apple OS X

En Apple OS X el registro de protocolo se realiza en el info.plist de la aplicación, con las siguientes líneas:

<array>

<dict>

<key>CFBundleURLSchemes</key>

<array>

<string>afirma</string>

</array>

<key>CFBundleURLName</key>

<string>Protocolo de invocación del Cliente @firma</string>

</dict>

</array>

Dado que en Apple OS X el info.plist es interno a cada aplicación, se elimina con esta (deshaciéndose el registro de protocolo).

El proceso no necesita permisos de administrador.

#### Linux

En Linux se realiza una asociación del protocolo mediante una doble declaración en el paquete DEB de instalación:

##### Declaración del x-scheme-handler en el desktop de Gnome

[Desktop Entry]

Encoding=UTF-8

Name=AutoFirma Cliente @firma

Comment=Cliente @firma

Exec=java -jar /usr/lib/autofirma/autofirma.jar

Icon=/usr/share/autofirma/autofirma.svg

MimeType=x-scheme-handler/afirma;

Terminal=false

Type=Application

Categories=GNOME;Application;Office

StartupNotify=true

StartupWMClass=autofirma

##### Declaración como preferencia de Mozilla Firefox en /etc/Firefox/pref

pref("network.protocol-handler.app.afirma","/usr/bin/simpleafirma");

pref("network.protocol-handler.warn-external.afirma",false);

pref("network.protocol-handler.external.afirma",true);

### Instalación de un certificado SSL auto firmado como de confianza para el sistema

El instalador, durante el proceso de instalación, genera un certificado para comunicaciones SSL asociado al host “127.0.0.1” (la dirección local).

Este certificado tiene las siguientes características:

* Se genera al vuelo durante el proceso de instalación.
* El cálculo de claves usa el generador de números aleatorios por defecto de Java 8.
* El tamaño de la clave es de 2048 bits.

Este certificado, una vez generado:

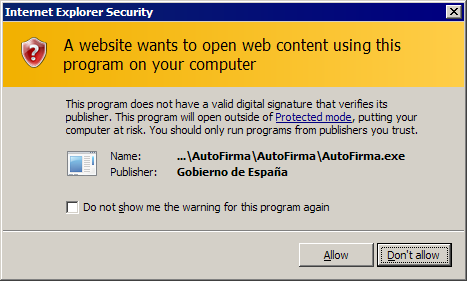
1. Se instala en el sistema como de confianza para servidores SSL.
   1. En Mozilla Firefox usando la herramienta “CertUtil” de Mozilla.
   2. En MS-Windows usando las funciones de Java 8 de acceso a CAPI.
   3. En OS X usando las funciones de Java 8 de acceso al llavero de OS X.
2. Se guarda como PKCS#12 en el directorio de activos del programa AutoFirma ($HOME/.afirma, donde $HOME es el directorio del usuario) con una contraseña prefijada (en el propio código del aplicativo).

### Otros cambios

El programa de instalación puede crear iconos en el menú inicio o el escritorio del usuario, así como dar de alta el desinstalador (por ejemplo, en el caso de Windows, para poder desinstalar la aplicación desde la sección “Programas y Características” del Panel de Control), pero ningún cambio que modifique o afecte a la seguridad del equipo del usuario firmante.

## Seguridad en el propio binario ejecutable de AutoFirma

Los binarios de AutoFirma están firmados digitalmente en el caso de Microsoft Windows. Internet Explorer sobre Microsoft Windows es el único navegador Web que comprueba las firmas electrónicas de los programas que se han arrancado mediante una invocación por protocolo, y es la razón por la que es el binario de Microsoft Windows el único firmado.



No se firman ni los binarios de OS X o Linux ni los JAR de Java internos a la aplicación AutoFirma debido a que esta firma nunca se advierte de su comprobación al usuario por protocolo (por lo que carece de utilidad), pero si puede ocasionar retardos considerables en la carga de la aplicación, puesto que internamente el sistema operativo si calcula las huellas digitales e incluso consulta a los sistemas OCSP.

## Seguridad en el propio proceso de firma

El proceso de firma implementa distintas medidas de seguridad, que van implícitas en la secuencia de uso:

1. El navegador Web invoca por protocolo (afirma://…) a la aplicación AutoFirma.
   1. La invocación por protocolo no puede ser interceptada por un programa espía de conexiones de red, ya que no se realiza mediante TCP/IP ni UDP.
   2. En la URL de invocación se indica (entre otras cosas):
      1. Una lista acotada de puertos aleatorios TCP de conexión, generados dentro del rango de puertos para uso dinámico (desde el 49152 al 65535).

De esta forma, el servicio se inicia cada vez en un puerto distinto y aleatorio, dificultando ataques previamente planificados a puertos concretos.

* + 1. Un identificador de sesión aleatorio generado localmente al vuelo desde JavaScript. El identificador se generará usando las funciones de aleatorios de la “*Web Cryptography API*” si el motor JavaScript lo implementa o un algoritmo basado en la generación de aleatorios corriente y marcas de tiempo en caso de no hacerlo.

1. El programa AutoFirma se inicia y abre un servicio servidor de tipo socket TCP SSL en uno de los puertos posibles de la lista recibida, respondiendo a un subconjunto del protocolo HTTPS.
   1. El servicio no es concurrente, una vez una instancia de AutoFirma acepta una conexión entrante en su puerto, ignora cualquier otra petición de conexión.
2. El programa AutoFirma acepta la conexión SSL desde el JavaScript que ejecuta localmente el navegador Web.
   1. Solo se aceptan conexiones que provengan desde el mismo equipo local (localhost / 127.0.0.1). Cualquier conexión externa es rechazada.
   2. Solo se aceptan conexiones que indiquen como parámetro en el diálogo el ID de sesión, que al haber sido generado al vuelo localmente y de forma aleatoria por el JavaScript nadie más puede conocer.
3. El programa realiza la firma electrónica, para lo cual **siempre** habrá algún tipo de confirmación visual de cara al usuario firmante (usualmente, el mismo diálogo de selección de certificados).
4. Una vez termina el proceso (o pasa cierto tiempo sin intercambiarse mensajes), el programa se cierra, cerrándose a la vez el servicio SSL.
   1. El hecho de tener el servicio abierto únicamente el tiempo necesario reduce la ventana de tiempo en la que este puede recibir ataques.
   2. Si se volviéndose a requerir la aplicación, se invocará nuevamente por protocolo proporcionándole nuevos puertos de conexión para el socket y un nuevo ID de sesión.