Guía Completa de Usuario - S-FiDE: Sistema de Firma Digital Extendida - v1.0.0

Índice

PDFSignerPKCS11

Descripción

Aplicación para firmar documentos PDF utilizando un token criptográfico (dispositivo PKCS#11). Permite agregar firmas digitales a documentos PDF utilizando certificados almacenados en dispositivos de hardware como tokens USB o smart cards. Implementa el estándar PAdES (PDF Advanced Electronic Signatures) conforme a ETSI EN 319 142.

Ejecución: `bash java -jar PDFSignerPKCS11.jar -i <archivo.pdf> -l <lib-pkcs11.so> -p <password> -s <slot> [-k true/false] [-x pos] [-y pos] [-t "texto"]`

Parámetros de Entrada

Parámetros Obligatorios

- `-i, --input` : Ruta al archivo PDF a firmar
- `-l, --library` : Ruta a la biblioteca PKCS#11 del token
- `-p, --password` : Contraseña del token
- `-s, --slot` : Número de slot del token

Parámetros Opcionales

- `-k, --lock` : Bloquear el documento después de firmar (true/false)
- `-x, --xpos` : Posición X de la firma visible
- `-y, --ypos` : Posición Y de la firma visible
- `-t, --text` : Texto personalizado para la firma visible

Comandos Especiales

- `-v, --version` : Muestra la versión del programa
- `-h, --help` : Muestra la ayuda
- `--license` : Muestra la licencia

Validaciones y Estándares

- Cumple con el estándar PAdES (ETSI EN 319 142)
- Soporta firmas con sello de tiempo (timestamp)
- Verificación de certificados mediante OCSP
- Verificación de integridad del documento
- Soporte para múltiples firmas
- Soporta algoritmos SHA-256 para el digest y RSA para la firma
- Compatible con el estándar PKCS#11 v2.20

Salida

- Genera un nuevo archivo PDF firmado con el sufijo "-signed"
- La ubicación del archivo de salida se muestra en consola
- Si se especifican coordenadas (x,y), agrega una firma visible en el documento

Mensajes de Error

- "No se pudo configurar UTF-8 para la salida"

- "El archivo PDF no existe o no es accesible"
- "La biblioteca PKCS#11 no existe o no es accesible"
- "El PDF está encriptado y no puede ser firmado"
- "La firma existente '[nombre]' no es válida"
- "El token no contiene una clave privada válida"
- "No se encontró una cadena de certificados válida en el token"
- "Error al acceder a la clave privada del token"

PDFSignerPKCS12

Descripción

Aplicación para firmar documentos PDF utilizando certificados digitales en formato PKCS#12 (.p12 o .pfx). Permite agregar firmas digitales a documentos PDF utilizando certificados almacenados en archivos. Implementa el estándar PAdES (PDF Advanced Electronic Signatures) conforme a ETSI EN 319 142.

Ejecución: `bash java -jar PDFSignerPKCS12.jar -i <archivo.pdf> -c <certificado.p12> -p <password> [-k true/false] [-x pos] [-y pos] [-t "texto"]`

Parámetros de Entrada

Parámetros Obligatorios

- `-i, --input` : Ruta al archivo PDF a firmar
- `-c, --certificate` : Ruta al archivo del certificado PKCS#12
- `-p, --password` : Contraseña del certificado

Parámetros Opcionales

- `-l, --lock` : Bloquear el documento después de firmar (true/false)
- `-x, --xpos` : Posición X de la firma visible
- `-y, --ypos` : Posición Y de la firma visible
- `-t, --text` : Texto personalizado para la firma visible

Comandos Especiales

- `-v, --version` : Muestra la versión del programa
- `-h, --help` : Muestra la ayuda

Validaciones y Estándares

- Cumple con el estándar PAdES (ETSI EN 319 142)
- Soporta firmas con sello de tiempo (timestamp)
- Verificación de certificados mediante OCSP
- Verificación de integridad del documento
- Soporte para múltiples firmas
- Soporta algoritmos SHA-256 para el digest y RSA para la firma
- Compatible con el estándar PKCS#12 v1.1

Salida

- Genera un nuevo archivo PDF firmado con el sufijo "-signed"
- La ubicación del archivo de salida se muestra en consola
- Si se especifican coordenadas (x,y), agrega una firma visible en el documento

Mensajes de Error

- "El archivo PDF no existe o no es accesible"
- "El archivo de certificado no existe o no es accesible"
- "El PDF está encriptado y no puede ser firmado"

- "La firma existente '[nombre]' no es válida"
- "El certificado no contiene una clave privada"
- "No se encontró una cadena de certificados válida"
- "Error al acceder a la clave privada"

PDFVerifySignatures

Descripción

Aplicación para verificar la validez de las firmas digitales en documentos PDF. Realiza validaciones de integridad, autenticidad y estado de revocación de los certificados utilizados en las firmas. Compatible con firmas PAdES y formatos PDF/A.

Ejecución: `bash java -jar PDFVerifySignatures.jar <archivo.pdf> [-simple]`

Parámetros de Entrada

Parámetros Obligatorios

- Ruta al archivo PDF a verificar (primer argumento)

Parámetros Opcionales

- `-simple`: Muestra una salida simplificada de la verificación

Comandos Especiales

- `-version`: Muestra la versión del programa
- `-licencia`: Muestra la licencia
- `-ayuda`: Muestra la ayuda

Validaciones y Estándares

- Verifica firmas PAdES conforme a ETSI EN 319 142
- Validación de sellos de tiempo (timestamp)
- Verificación de certificados mediante OCSP y CRL
- Verificación de integridad del documento
- Comprobación de la cadena de certificación
- Validación de firmas LTV (Long Term Validation)
- Soporte para múltiples firmas en el mismo documento

Salida

- Estado de cada firma en el documento
- Información detallada de cada firma (a menos que se use -simple):
  - Firmante
  - Organización
  - Número de serie del certificado
  - Fechas de validez
  - Emisor
  - Algoritmo de firma
  - Cadena de certificación
- Estado final del documento (válido/inválido)
- Estado de bloqueo y encriptación del documento

Mensajes de Error

- "El documento no contiene firmas digitales"
- "Error verificando firma [nombre]"
- "DOCUMENTO INVÁLIDO: Una o más firmas no son válidas"
- "Certificado revocado al momento de la firma"
- "Certificado no confiable o autofirmado"

PKCS12CertificateExtractor

Descripción

Herramienta para extraer certificados digitales de archivos PKCS#12 (.p12 o .pfx) y exportarlos en formato PEM. Permite la extracción segura de certificados para su uso en otros sistemas o verificaciones.

Ejecución: `bash java -jar PKCS12CertificateExtractor.jar <archivo.p12> <password>`

Parámetros de Entrada

Parámetros Obligatorios

- Ruta al archivo PKCS#12 (primer argumento)
- Contraseña del archivo PKCS#12 (segundo argumento)

Comandos Especiales

- `-version`: Muestra la versión del programa
- `-licencia`: Muestra la licencia
- `-ayuda`: Muestra la ayuda

Validaciones y Estándares

- Compatible con el estándar PKCS#12 v1.1
- Soporta certificados X.509 v3
- Manejo de codificación UTF-8
- Validación de integridad del archivo PKCS#12
- Extracción de cadena completa de certificación

Salida

- Información del certificado:
  - Sujeto
  - Emisor
  - Número de serie
  - Período de validez
  - Algoritmo de firma
- Archivo PEM con el certificado extraído

Mensajes de Error

- "El archivo PKCS#12 no existe"
- "El archivo no es un PKCS#12 válido o la contraseña es incorrecta"
- "No se encontró ningún certificado X.509 en el archivo PKCS#12"
- "Error al exportar certificado"

TokenCertificateExtractor

Descripción

Herramienta para extraer certificados digitales de tokens criptográficos (dispositivos PKCS#11) y exportarlos en formato PEM. Permite la obtención de certificados almacenados en dispositivos de hardware de forma segura.

Ejecución: `bash java -jar TokenCertificateExtractor.jar <lib-pkcs11.so> <password> <slot>`

Parámetros de Entrada

Parámetros Obligatorios

- Ruta a la biblioteca PKCS#11 (primer argumento)
- Contraseña del token (segundo argumento)
- Número de slot (tercer argumento)

Comandos Especiales

- `-version`: Muestra la versión del programa
- `-licencia`: Muestra la licencia
- `-ayuda`: Muestra la ayuda

Validaciones y Estándares

- Compatible con el estándar PKCS#11 v2.20
- Soporta certificados X.509 v3
- Manejo de codificación UTF-8
- Validación de acceso al token
- Extracción de cadena completa de certificación

Salida

- Información del certificado:
  - Sujeto
  - Emisor
  - Número de serie
  - Período de validez
  - Algoritmo de firma
- Archivo PEM con el certificado extraído

Mensajes de Error

- "El archivo de la biblioteca PKCS#11 no existe"
- "Proveedor SunPKCS11 no disponible"
- "Error al cargar el almacén de claves"
- "No se encontró ningún certificado en el slot [número]"
- "Error al exportar el certificado"

TokenSlotsView

Descripción

Herramienta para visualizar y listar los slots disponibles en un token criptográfico, incluyendo la información de los certificados almacenados en cada slot. Útil para diagnóstico y verificación de tokens PKCS#11.

Ejecución: `bash java -jar TokenSlotsView.jar <lib-pkcs11.so> <password>`

Parámetros de Entrada

Parámetros Obligatorios

- Ruta a la biblioteca PKCS#11 (primer argumento)
- Contraseña del token (segundo argumento)

Comandos Especiales

- `-version`: Muestra la versión del programa
- `-licencia`: Muestra la licencia
- `-ayuda`: Muestra la ayuda

Validaciones y Estándares

- Compatible con el estándar PKCS#11 v2.20
- Soporte para múltiples slots
- Identificación de tipos de certificados
- Validación de objetos del token
- Verificación de estado de certificados

Salida

Para cada slot encontrado, muestra: - Número de slot - Alias del certificado - Tipo (Clave Privada o Certificado) - Información del certificado: - Sujeto - Emisor - Período de validez - Número de serie

Mensajes de Error

- "El archivo de la biblioteca PKCS#11 no existe"
- "El proveedor SunPKCS11 no está disponible"
- "Contraseña incorrecta o error al acceder al token"
- "Error al leer el token"
- "No se encontraron certificados ni claves en el token"

XMLSignerPKCS11

Descripción

Aplicación para firmar documentos XML utilizando un token criptográfico (dispositivo PKCS#11). Permite firmar documentos XML completos o elementos específicos dentro del documento, cumpliendo con los estándares XMLDSig y XAdES.

Ejecución: `bash java -jar XMLSignerPKCS11.jar <lib-pkcs11.so> <password> <slot> <archivo.xml> <elemento_xml>`

Parámetros de Entrada

Parámetros Obligatorios

- Ruta a la biblioteca PKCS#11 (primer argumento)
- Contraseña del token (segundo argumento)
- Número de slot (tercer argumento)
- Ruta al archivo XML (cuarto argumento)
- Párrafo o elemento XML a firmar (quinto argumento) - Usar string vacío para firmar todo el documento ("")

Comandos Especiales

- `-version` : Muestra la versión del programa
- `-licencia` : Muestra la licencia
- `-ayuda` : Muestra la ayuda

Validaciones y Estándares

- Compatible con XMLDSig (XML Digital Signature)
- Soporta firmas XAdES (XML Advanced Electronic Signatures)
- Implementa transformaciones de canonicalización XML
- Soporta firmas enveloped y detached
- Validación de párrafor o elementos XML a firmar, URIs y referencias
- Algoritmo de firma RSA-SHA256
- Soporte para namespaces XML
- Preservación de la estructura del documento

Salida

- Genera un nuevo archivo XML firmado con el sufijo "-signed"
- La ubicación del archivo de salida se muestra en consola

Mensajes de Error

- "El archivo de la biblioteca PKCS#11 no existe"
- "El archivo XML no existe"
- "El elemento o párrafo XML especificado no existe en el documento XML"
- "Error en el acceso al token"
- "Proveedor SunPKCS11 no disponible"
- "Contraseña incorrecta"
- "No se encontró el elemento XML con identificador"

XMLSignerPKCS12

Descripción

Aplicación para firmar documentos XML utilizando certificados digitales en formato PKCS#12 (.p12 o .pfx). Permite firmar documentos XML completos o elementos específicos dentro del documento, cumpliendo con los estándares XMLDSig y XAdES.

Ejecución: `bash java -jar XMLSignerPKCS12.jar <certificado.p12> <password> <archivo.xml> <elemento_xml>`

Parámetros de Entrada

Parámetros Obligatorios

- Ruta al archivo PKCS#12 (primer argumento)
- Contraseña del certificado (segundo argumento)
- Ruta al archivo XML (tercer argumento)
- Párrafo o elemento XML a firmar (cuarto argumento) - Usar string vacío para firmar todo el documento ("")

Comandos Especiales

- `-version` : Muestra la versión del programa
- `-licencia` : Muestra la licencia
- `-ayuda` : Muestra la ayuda

Validaciones y Estándares

- Compatible con XMLDSig (XML Digital Signature)
- Soporta firmas XAdES (XML Advanced Electronic Signatures)
- Implementa transformaciones de canonicalización XML
- Soporta firmas enveloped y detached
- Validación de párrafos o elementos XML a firmar, URIs y referencias
- Algoritmo de firma RSA-SHA256
- Soporte para namespaces XML
- Preservación de la estructura del documento

Salida

- Genera un nuevo archivo XML firmado con el sufijo "-signed"
- La ubicación del archivo de salida se muestra en consola

Mensajes de Error

- "El archivo PKCS#12 no existe"
- "El archivo XML no existe"
- "El archivo no es un PKCS#12 válido o la contraseña es incorrecta"
- "El archivo PKCS#12 no contiene ningún certificado"
- "El elemento o párrafo XML especificado no existe en el documento XML"
- "No se encontró el elemento XML con identificador"

XMLVerifySignatures

Descripción

Aplicación para verificar la validez de las firmas digitales en documentos XML. Realiza validaciones de integridad, autenticidad y estado de revocación de los certificados utilizados en las firmas, compatible con XMLDSig y XAdES.

Ejecución: `bash java -jar XMLVerifySignatures.jar <archivo.xml> [-simple]`

Parámetros de Entrada

Parámetros Obligatorios

- Ruta al archivo XML (primer argumento)

Parámetros Opcionales

- `-simple`: Muestra una salida simplificada de la verificación

Comandos Especiales

- `-version`: Muestra la versión del programa
- `-licencia`: Muestra la licencia
- `-ayuda`: Muestra la ayuda

Validaciones y Estándares

- Verificación de firmas XMLDSig y XAdES
- Validación de certificados mediante OCSP y CRL
- Verificación de integridad de referencias XML
- Soporte para múltiples firmas
- Validación de transformaciones XML
- Verificación de sellos de tiempo
- Comprobación de la cadena de certificación

Salida

- Estado de cada firma en el documento
- Información detallada de cada firma (a menos que se use -simple):
  - Estado de validez
  - Método de canonicalización
  - Método de firma
  - Valor de la firma
  - Información del certificado
  - Estado de revocación
- Resultado final de la validación

Mensajes de Error

- "El archivo XML no existe"
- "El documento XML no contiene firmas digitales"
- "Error al procesar el archivo XML"

- "DOCUMENTO INVÁLIDO: Una o más firmas contienen errores graves"
- "No se encontró KeyInfo"
- "No se encontró una clave válida"

XMLVerifyXSDStructure

Descripción

Herramienta para validar documentos XML contra su esquema XSD y verificar sus firmas digitales. Puede utilizar un esquema XSD local o descargar automáticamente el esquema referenciado en el documento XML.

Ejecución: `bash java -jar XMLVerifyXSDStructure.jar <archivo.xml> [esquema.xsd]`

Parámetros de Entrada

Parámetros Obligatorios

- Ruta al archivo XML (primer argumento)

Parámetros Opcionales

- Ruta al archivo XSD (segundo argumento)

Comandos Especiales

- `-version`: Muestra la versión del programa
- `-licencia`: Muestra la licencia
- `-ayuda`: Muestra la ayuda

Validaciones y Estándares

- Validación contra XML Schema 1.1
- Soporte para XMLDSig y XAdES
- Resolución automática de esquemas externos
- Validación de espacios de nombres
- Verificación de tipos de datos XML
- Soporte para restricciones y patrones XSD
- Validación de referencias cruzadas
- Procesamiento seguro de entidades externas

Salida

- Resultado de la validación contra el esquema XSD
- Verificación de firmas digitales
- Estado final del documento
- Advertencias si el XSD proporcionado difiere del referenciado en el XML

Mensajes de Error

- "El archivo XML no existe"
- "El archivo XSD no existe"
- "No se encontró referencia a esquema XSD en el XML"
- "Error al procesar el archivo XSD"
- "Error de validación XML"
- "El documento XML no contiene firmas digitales"
- "Se encontraron errores en la validación del documento XML"