Manual de usuario
Mastin v1.0 y XMastin v1.0

Índice

Descripción

Mastin v1.0

Aplicación de línea de comandos que encripta o desencripta ficheros a partir de la clave extraida de un KeyStore creado por la herramienta ***keytool*** de Java
El algoritmo que se usa es el algoritmo asimétrico RSA inventado por Rivest, Adleman y Shamir y que viene recogido en el PKCS#1.
El modo de funcionamiento será ECB (Electronic Code Book) y el relleno será OAEP (Optimal Asymmetric Encryption Padding) definido en el PKCS#1 v2.
La clave, ya sea pública o privada, se extraerá de la entrada en un KeyStore definida por un alias
Se podrá especificar la ruta del KeyStore (*-KeyStorePath keyStorePath*), su tipo (*-KeyStoreType keyStoreType*), y su clave (*-KeyStorePassword keyStorePassword*).
Si se desea utilizar un alias distinto a "mykey", se deberá especificar (*-Alias alias*) así como su clave (*-KeyPassword keyPassword*) en caso a que sea distinta a la del KeyStore (*-KeyStoreType keyStoreType*)
Al estar escrito en Java, se puede ejecutar en cualquier Sistema Operativo, siendo testados tanto en Windows (XP) como en Linux (SuSE Linux 8.0).

XMastin v1.0

Aplicación para un entorno gráfico que utiliza el API suministrado por Mastin para realizar las mismas funciones que éste.

Al estar escrito en Java, se puede ejecutar en cualquier Sistema Operativo, siendo testados tanto en Windows (XP) como en Linux (SuSE Linux 8.0).

Tanto Mastin como XMastin fueron creadas por Gabriel Babiano Huete (gabrielbabiano@yahoo.es) con motivo de su Proyecto Fin de Carrera de Ingeniería de Telecomunicación en la Escuela Superior de Ingenieros de la Universidad de Sevilla en 2002.

Software necesario

Tanto para la ejecución de Mastin como XMastin se necesitan tener el siguiente software correctamente instalado:

- Java 2 v1.4.0 JRE (Java Runtime Enviroment) o SDK (Standard Development Kit) o posterior
- jce_policy-1_4_0.zip
- JCE Provider 1.15 de Bouncy Castle (se instala automáticamente en la instalación de Mastin)

No debemos olvidar que dado que XMastin se basa en el API que suministra Mastin, requerirá además que este último se encuentre correctamente instalado

---

Instalación

Usuarios de Windows

Para instalar Mastin v1.0 y XMastin v1.0 deberá de tener instalado previamente Java 2 v1.4.0 pudiendo elegir entre solamente el JRE (Java Runtime Enviroment) o el SDK (Standard Development Kit) completo. Deberemos posteriormente instalar jce_policy-1_4_0.zip. Posteriormente deberá ejecutar "instalarMastin.bat". Dicho batch copiará automáticamente:

- el directorio "Mastin_v1.0" recursivamente en "c:\archivos de programa\Mastin_v1.0\"
- el direcotrio "crypto-115" recursivamente en "c:\archivos de programa\crypto-115\"
- el batch "XMastin.bat" en %windir%
- el acceso directo "XMastin v1.0.lnk" en el escritorio
- el acceso directo "XMastin v1.0.lnk" en el directorio XMastin v1.0 creado en "Menú de inicio|Programas" ("Menú de inicio|Programas|XMastin v1.0")
- el acceso directo "ayuda.lnk" en "Menú de inicio|Programas|XMastin v1.0"
- el acceso directo "desinstalar Mastin v1.0 y XMastin v1.0" en "Menú de inicio|Programas"

En esta instalación también se está instalando el proveedor de JCE suministrado por Bouncy Castle

Usuarios de Linux

Para instalar Mastin v1.0 y XMastin v1.0 deberá de tener instalado previamente Java 2 v1.4.0 pudiendo elegir entre solamente el JRE (Java Runtime Enviroment) o el SDK (Standard Development Kit) completo. Deberemos posteriormente instalar jce_policy-1_4_0.zip Posteriormente deberá ejecutar "sh ./instalarMastin.sh". Dicho script copiará automáticamente:

- el directorio "Mastin_v1.0" recursivamente en "/usr/share/Mastin_v1.0"
- el directorio "crypto-115" recursivamente en "/usr/share/crypto-115"
- el script Mastin_v1.0/bin/XMastin.sh en /bin/
- el acceso directo "XMastin v1.0" en el escritorio "$HOME/Desktop/". Dicho acceso directo apunta al script "/bin/XMastin.sh"

- el acceso directo "desinstalar Mastin v1.0 y XMastin v1.0" en el escritorio "$HOME/Desktop/". Dicho acceso directo apunta al script "/usr/share/Mastin_v1.0/bin/desinstalarMastin.sh"

En esta instalación también se está instalando el proveedor de JCE suministrado por Bouncy Castle

---

Keytool

Sysnopsis

```
keytool [comandos]
```

Descripción

keytool es la herramienta de gestión de certificados y claves de Java. Gestiona una base de datos (keystore) de claves privadas y certificados X.509 con claves públicas autenticadas por entidades de confianza. El keystore se protege por una contraseña.

Entradas

Existen dos tipos de entradas en un keystore:

- ***keyEntry***: almacena entre otras cosas la clave privada por lo que se almacena de forma protegida con otra contraseña para prevenir accesos no autorizados.
- ***trustedCertEntry***: contiene una clave pública certificada. Se denomina ?trusted? (?de confianza?) porque el dueño del keystore confía que la clave pública del certificado pertenece realmente al ?subject? (?sujeto?) dueño del certificado. El expendedor del certificado se responsabiliza de esto firmando el certificado.

Alias

A todas las entradas del keystore se accede mediante un único alias. Los alias no distinguen mayúsculas de minúsculas.
El alias se especifica cuando se añade una entrada al keystore usando -genkey para generar un par de claves pública/privada o cuando se añade un certificado mediante el comando -import a la lista de certificados de confianza. Posteriores llamadas a keytool usarán el mismo alias para referirse a dicha entrada.

Localización del KeyStore

Cada comando keytool tiene la opción -keystore para especificar el nombre y localización del archivo persistente que contiene el keystore. Este archivo se almacena por defecto en un fichero llamado .keystore situado en el directorio del usuario determinado por la propiedad del sistema user.home.

Creación del KeyStore

Se crea un KeyStore cuando se usa cualquiera de los comandos -genkey, -import o -identitydb para añadir datos a un keystore que no existe aún.
Más específicamente, si no se especifica la opción -keystore y no existe aún un keystore, se creará uno llamado .keystore en el directorio determinado por user.home.

Implementación del KeyStore

La clase KeyStore del paquete java.security proporciona un interfaz bien definido para acceder y modificar la información contenida en un keystore. Se peden disponer de distintas implementaciones, cada una de un tipo determinado.
Existe una implementación propiedad de Sun Microsystems denominada JKS (Java KeyStore). Protege cada clave privada con una contraseña individual y también protege la integridad del keystore entero con otra contraseña (que puede ser igual a la anterior).
Por defecto se utiliza el tipo de keystore especificado explícitamente en la propiedad keystore.type en el archivo de propiedades de seguridad (normalmente es JKS). Dicho archivo se llama java.security y reside en java.home\lib\security donde java.home es el directorio del JRE.

Algoritmos soportados y tamaños de claves

keytool permite a los usuarios especificar el algoritmo de creación de pares de claves o de firma de cualquier proveedor de servicios criptográficos registrado. Es decir, las opciones -keyalg y -sigalg deben ser soportadas por la implementación del proveedor.

Generando pares de claves

Para generar un par de claves pública/privada, usaremos el comando -genkey de la forma:

```
gab:~ # keytool -genkey -keyalg RSA -keysize 2048
Enter keystore password: miContrasena
What is your first and last name?
[Unknown]: Gabriel Babiano Huete
What is the name of your organizational unit?
[Unknown]: Departamento de Telematica
What is the name of your organization?
[Unknown]: Universidad de Sevilla
What is the name of your City or Locality?
[Unknown]: Sevilla
What is the name of your State or Province?
[Unknown]: Spain
What is the two-letter country code for this unit?

[Unknown]: ES
Is correct?
[no]: y

Enter key password for
(RETURN if same as keystore password):
```

Crea una pareja de claves pública/privada para el algoritmo RSA de tamaño 2048 bits en el alias por defecto mykey en el keystore por defecto .keystore situado el el directorio casa del usuario.

Generando una CSR

Para crear una Petición de Firmado de Certificado (CSR) usando el formato PKCS#10 se usará el comando -certreq. Con el siguiente ejemplo, generamos una CSR de la clave pública contenida en la entrada referenciada por el alias mykey y lo guardamos en el fichero csr.pem:

```
keytool -certreq -alias mykey -file csr.pem
```

Posteriormente, este CSR se podrá remitir a una CA, y ésta, nos remitirá dicha petición ya firmada en un certificado una vez que haya comprobado la identidad del solicitante.

Importando certificados

Para importar certificados desde un archivo usaremos el comando -import de la siguiente manera:

```
keytool -import -alias cert -file certfile.cer
```

que importa el certificado del archivo certfile.cer y lo guarda el en keystore en la entrada identificado por el alias cert.

Se puede importar un certificado por dos posibles motivos:

- para añadirlo a la lista de certificados de confianza
- para importar un certificado respuesta de una CA como resultado de enviarle una CSR (obtenida mediante -certreq) a dicha CA.

Dependiendo del tipo de entrada que sea la apuntada por el alias:

- Si el alias apunta a una keyEntry, entonces keytool asume que se está importando un certificado respuesta. keytool comprueba si la clave pública del certificado coincide con la guardada bajo el alias y se sale si son diferentes.
- Si el alias no apunta a un keyEntry, entonces keytool asume que se está añadiendo una entrada de certificado de confianza. En este caso, el alias no debería existir en el keystore. Si no fuera así, keytool mostraría un error y no importaría el certificado. Si el alias no existiese en el keystore, keytool crearía una entrada de certificado confiado con el alias especificado y lo asociaría con el certificado importado.

keytool puede importar certificados X.509 v1, v2 y v3 y cadenas de certificados en formato PKCS#7 consistentes en certificados de ese tipo. Los datos para ser importados deben ser provistos ya sea en formato codificado binario o en formato codificado imprimible (también conocido como Base64) definido en el standard RFC 1421. En el último caso, la codificación debe estar limitada por una cadena que empiece por -----BEGIN al principio y -----END al final.

Exportando certificados

Para exportar un certificado a un archivo se usa el comando -export de la forma:

```
keytool -export -alias cert -file certfile.cer
```

Este ejemplo exporta el certificado cert al archivo certfile.cer. Esto es, si el alias cert corresponde a un keyEntry, se exporta el certificado que autentica la clave pública de jane. Si por el contrario, cert es el alias de un trustedCertEntry, entonces se exporta el certificado confiado.

Mostrando certificados

Para mostrar el contenido de un keystore, se usa el comando -list, de la forma:

```
keytool -list
```

o de la forma

```
keytool -list -alias mykey
```

para mostrar el contenido de la entrada del keystore referida por mykey
Para mostrar el contenido de un certificado guardado en un archivo, se usa el comando -printcert de la forma:

```
keytool -printcert -file certfile.cer
```

que muestra la información contenida en el archivo certfile.cer. Para esto último no se necesita un keystore.

Borrando entradas

Para borrar entradas, ya sean keyEntry o trustedCertEntry, lo podemos hacer con el comando -delete. Con el siguiente ejemplo borramos la entrada asociada al alias mykey:

```
keytool -delete -alias mykey
```

Opciones

Opciones que aparecen para la mayoría de los comandos:

Opcion	Descripción
-v	“Verbose”. Se mostrará información detallada
-help	Se mostrará la ayuda
-storetype *storetype*	Especifica el tipo de keystore
-keystore *keystore*	Localización del keystore
-storepass *storepass*	Contraseña utilizada para proteger la integridad del keystore. storepass debe tener al menos 6 dígitos. Si no se especifica en la línea de comandos, se preguntará por ella.
-provider *provider*	Nombre de la clase del proveedor del servicio criptográfico

Opciones por defecto

```
-alias mykey
-keyalg DSA
-keysize 1024
-validity 90
-keystore el_archivo_de_nombre_.keystore_en_el_directorio_casa_del_usuario

-file stdin_si_se_lee_o_stdout_si_se_escribe
```

El algoritmo de firma (opción -sigalg) se deriva del algoritmo de la clave privada:

- si es DSA, entonces la firma es SHA1
- si es RSA, entonces la firma es MD5

Para más información acerca de keytool, se puede consultar la documentación suministrada por Java.

Uso

Mastin v1.0

Una vez realizada la instalación, podemos proceder a ejecutarlo por línea de comandos:

Mastin [opciones] -e/-d -in inFile -out outFile

- *-e*: encripta
- *-d*: desencripta
- *-in inFile*: inFile es la ruta del fichero de entrada
- *-out outFile*: outFile es la ruta del fichero de salida

Lista de opciones:

- *-about*: muestra información acerca de la aplicación
- *-Alias alias*: alias del la entrada en la KeyStore de donde se extraerá la clave privada. Por defecto se usará "mykey"

- *-KeyPassword keyPassword*: especifica la contraseña de la clave del alias. Por defecto se usará la misma clave que para el KeyStore (*-KeyStorePassword KeyStorePassword*)
- *-KeyStorePassword KeyStorePassword*: especifica la contraseña de entrada al KeyStore
- *-KeyStorePath KeyStorePath*: especifica la ruta del fichero KeyStore
- *-KeyStoreType KeyStoreType*: especifica el tipo de KeyStore. Por defecto se usará "JKS"
- *-v*: muestra información del desarrollo de la aplicación (equivalente a -verbose)
- *-verbose*: muestra información del desarrollo de la aplicación
- *-version*: muestra información de la versión de la aplicación

Ejemplos:

Para encriptar:

```
java -cp /usr/java/crypto-115/jars/bcprov-jdk14-
115.jar:/usr/java/j2sdk1.4.0_02/jre/lib/jsse.jar:./jars/
es.us.esi.bab.mastin.Mastin -e -in kk.cer -out kk.crypt -keystorepassword
miclave -alias cert -v
```

Para desencriptar:

```
java -cp /usr/java/crypto-115/jars/bcprov-jdk14-
115.jar:/usr/java/j2sdk1.4.0_02/jre/lib/jsse.jar:./jars/
es.us.esi.bab.mastin.Mastin -d -in kk.crypt -out kk2.cer -alias mykey -
keystorepassword miclave -v
```

Se recomienda que en caso de disponer de entorno gráfico, se ejecute XMastin ya que facilita el paso de parámetros e incluso dispone de un generador de líneas de comando equivalente

XMastin v1.0

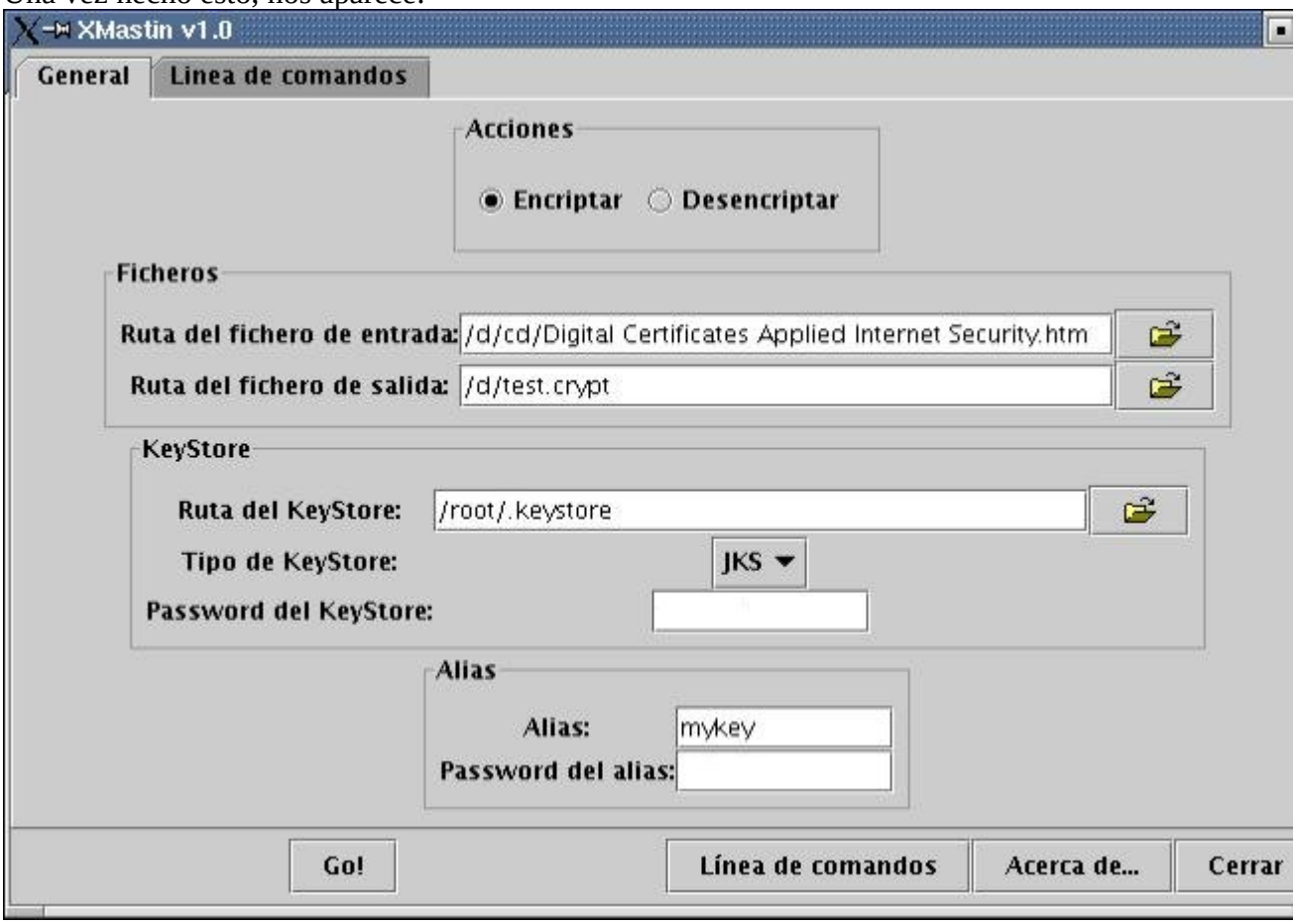
Una vez realizada la instalación, podemos proceder a ejecutarlo:

- en Windows podemos proceder haciendo doble click sobre los accesos directos a "XMastin v1.0" situados en el escritorio o bien en "Menú de inicio|Programas|XMastin v1.0" o bien ejecutando "XMastin" en la línea de comandos
- en Linux podemos entrarnos en la carpeta "XMastin v1.0" situada en el escritorio y una vez abierta, hacer doble click en el acceso directo a "XMastin v1.0" o bien en ejecutando "XMastin" en el shell

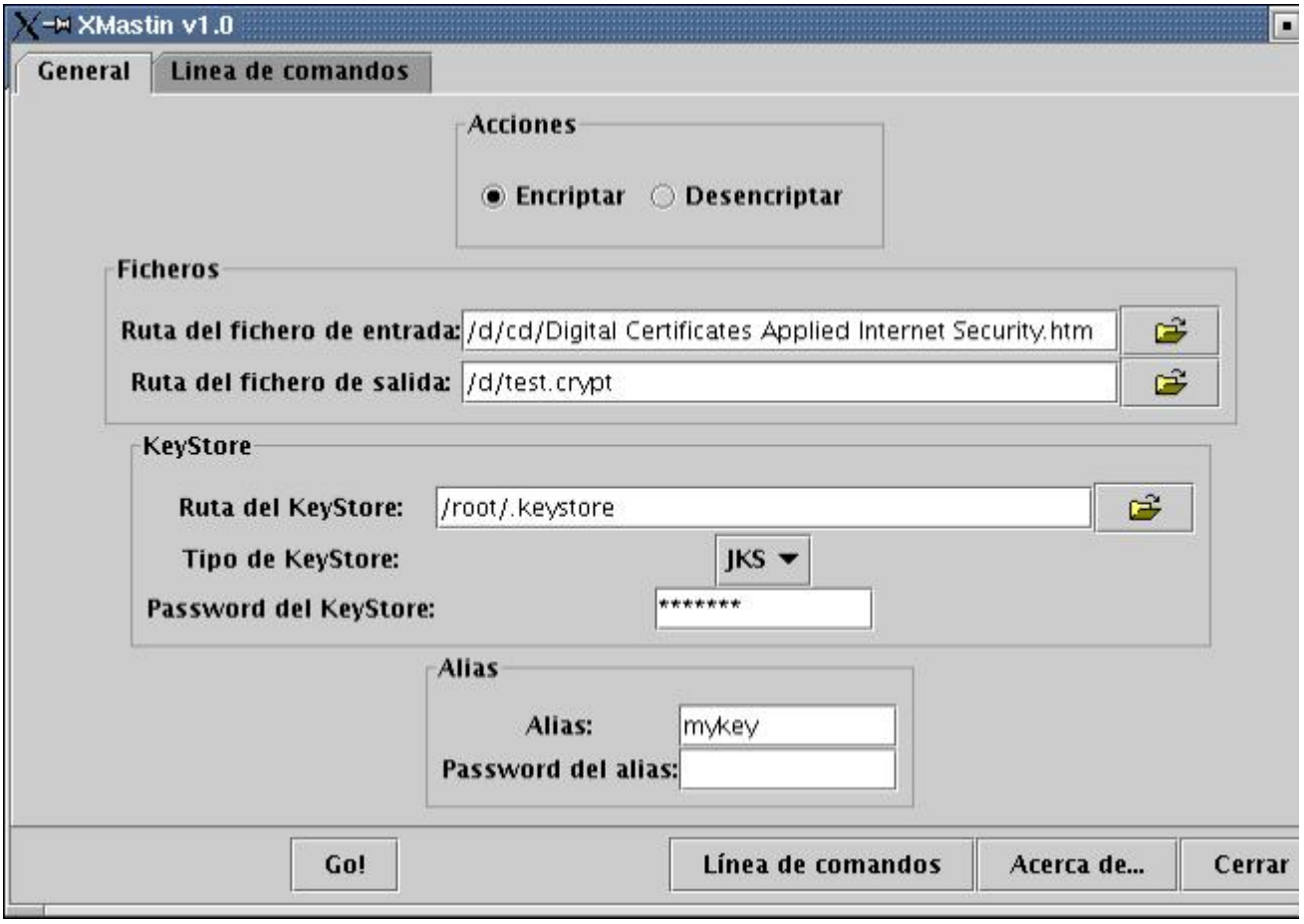
Una vez abierto, el programa, nos encontraremos con:

Aparecen dos pestañas: "General" y "Línea de comandos". En la pestaña "General", seleccionamos la acción que queremos realizar: encriptar o desencriptar. En este ejemplo vamos a encriptarlo, pero para desencriptarlo, se procedería de forma similar
Seleccionamos "Encriptar" en "Acciones"
Para especificar el fichero de entrada, tenemos dos opciones: escribirlo o seleccionarlo. Si deseamos seleccionarlo, pulsaremos el botón de la derecha:
Tras lo cual saldrá un diálogo similar a éste:

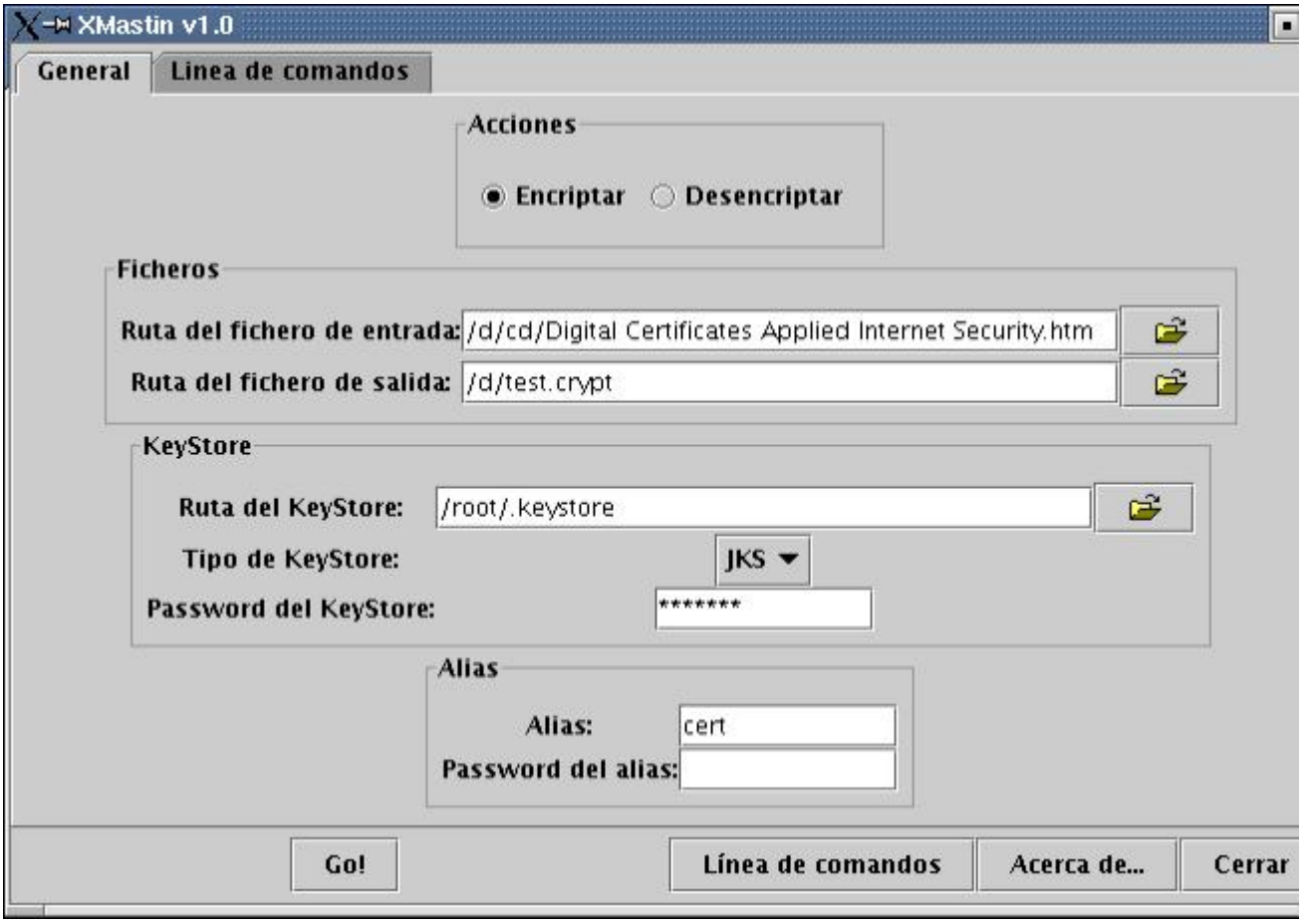
Seleccionaremos el archivo que deseemos y pulsaremos "Seleccionar" para seleccionarlo. Podemos pulsar cancelar en caso contrario:
Obraremos de la misma manera con el fichero de salida. Si el archivo ya existiese, se sobreescribirá
Una vez hecho esto, nos aparece:



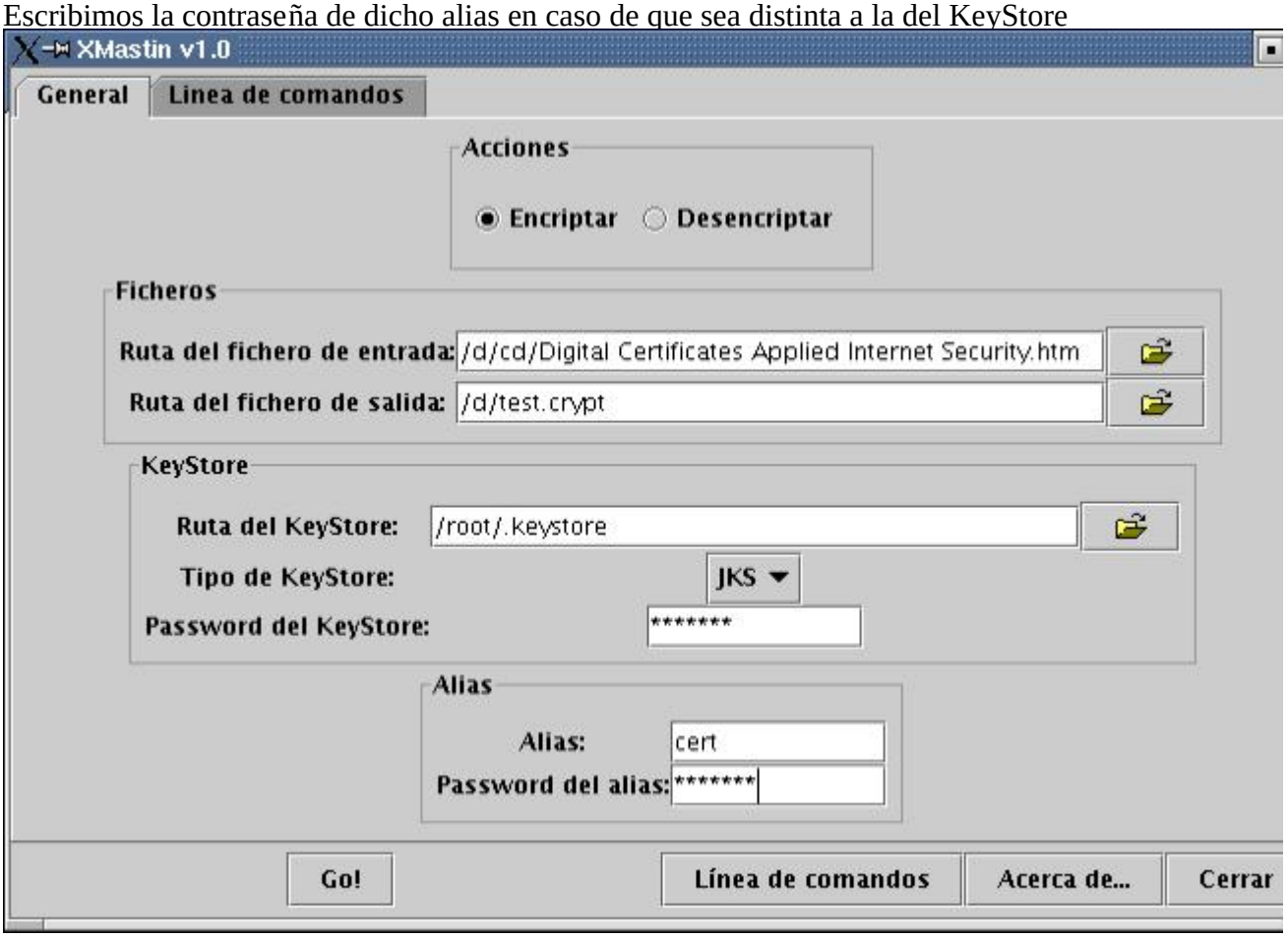
Si el KeyStore que se muestra es el adecuado (Mastín pone el KeyStore creado por defecto por *keytool*), introducimos la contraseña del mismo



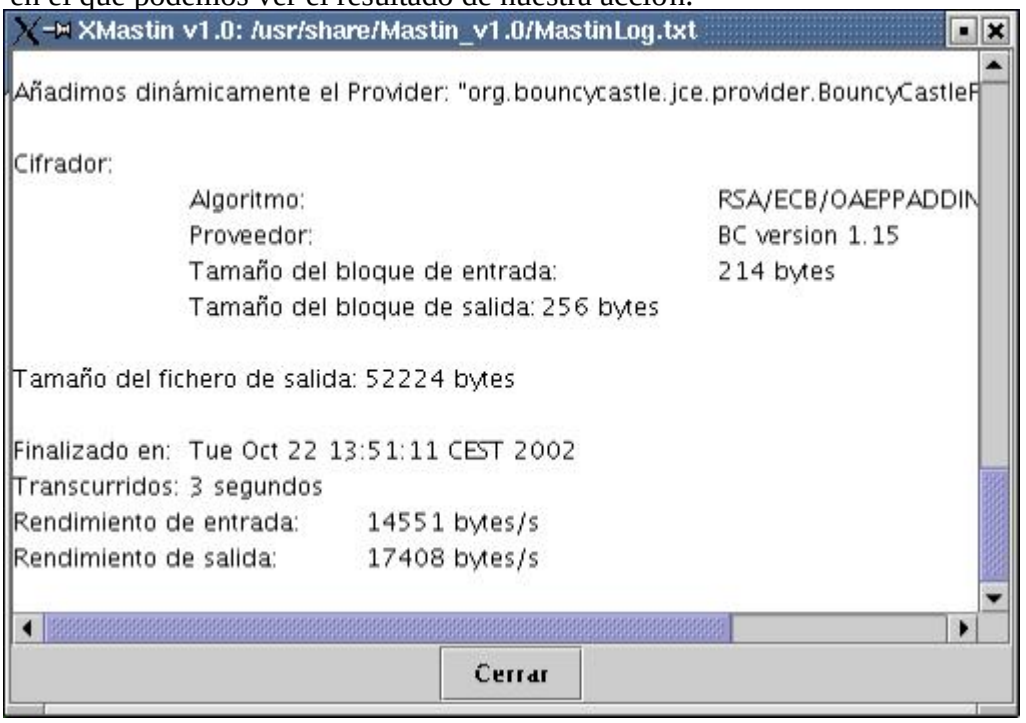
Cambiamos al alias que contiene la contraseña en el KeyStore con la que deseamos trabajar (Mastín pone por defecto el del alias que crea por defecto *keytool*). Si el alias se refiere a una entrada de tipo Key en el KeyStore, encriptaremos cun la clave privada. Si por el contrario, el alias se refiere a una entrada de Certificado en el Keytool, encriptaremos lógicamente con la clave pública



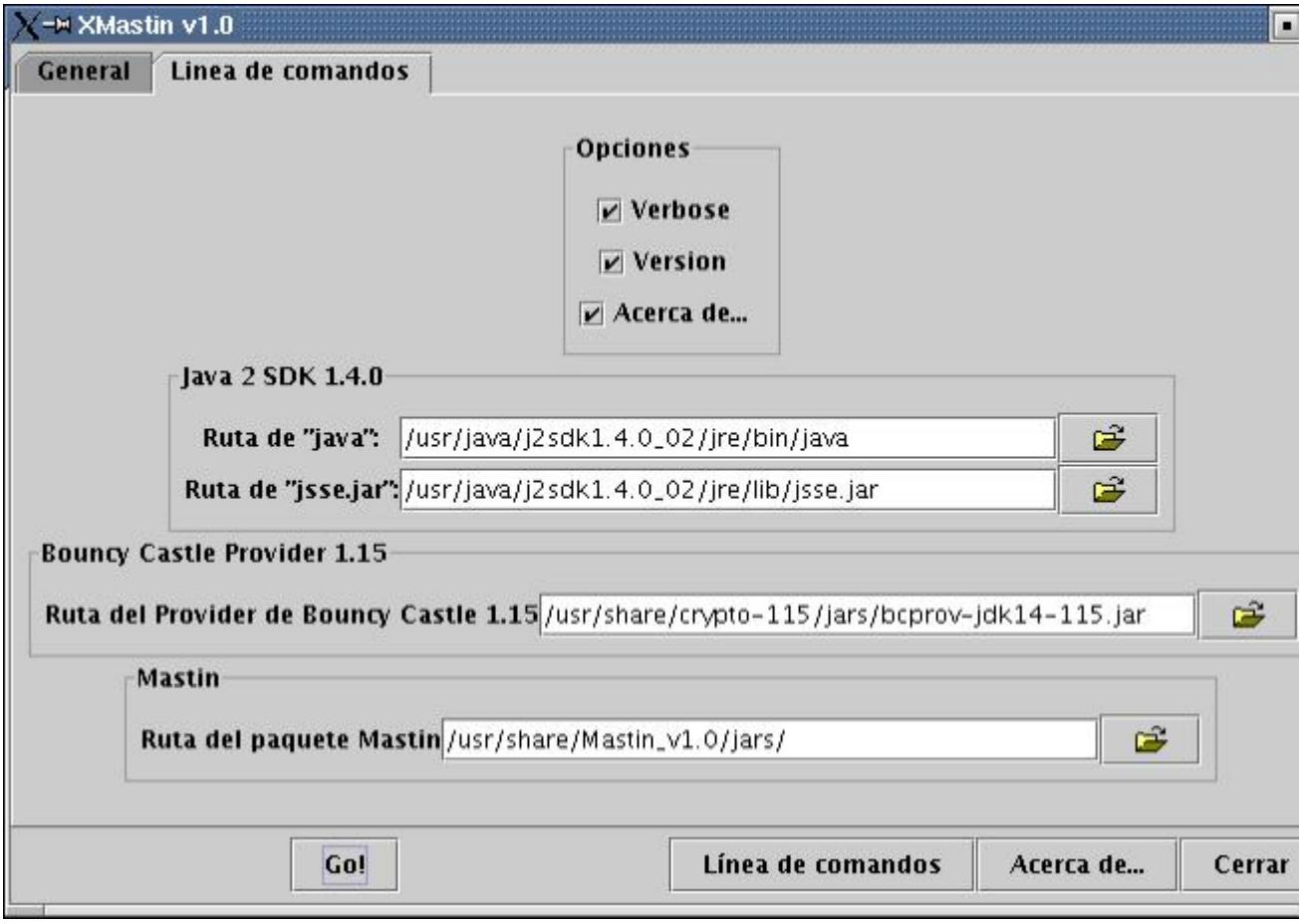
Escribimos la contraseña de dicho alias en caso de que sea distinta a la del KeyStore



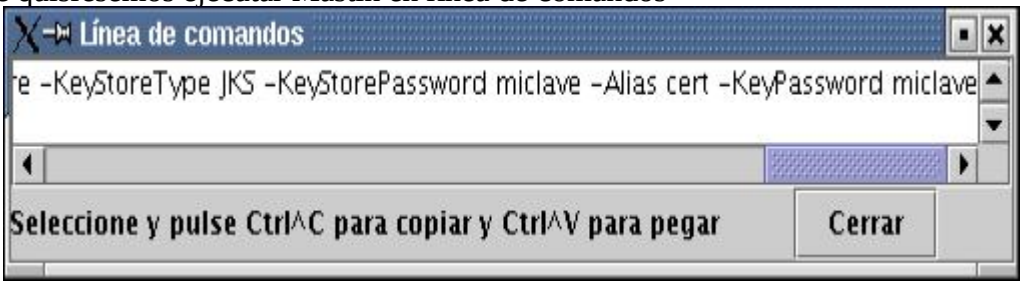
Pulsamos el botón de "Go!" situado en la parte inferior cuando ya hallamos introducidos los datos para que la aplicación se ponga a trabajar
Si los datos son correctos y hemos introducidos todos los necesarios, al cabo de algunos segundos (que dependen del tamaño del archivo de entrada y de si la clave es pública o privada) aparecerá una ventana que muestra el contenido del archivo "MastinLog.txt" situado en el mismo directorio de la aplicación; en el que podemos ver el resultado de nuestra acción.



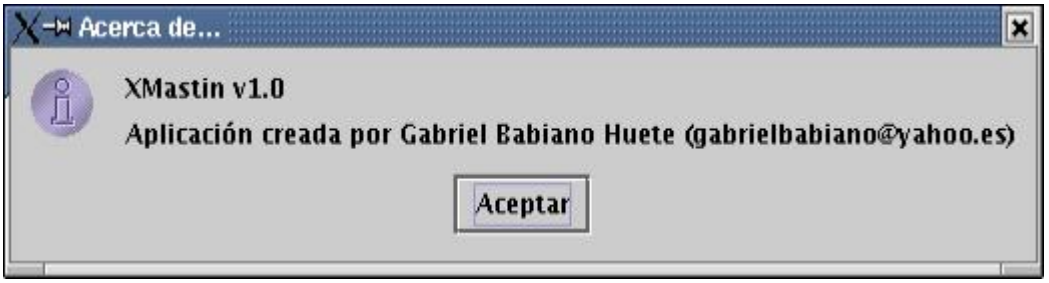
Para cerrarlo, pulsaremos el botón "Cerrar" situado en la parte inferior.
En la pestaña de "Línea de comandos" podemos cambiar las opciones para posteriormente obtener la línea de comandos equivalente a los datos que le hemos introducido en este formulario.



El el recuadro de opciones, podemos cambiar el verbose (si deseamos o no, que Mastín nos muestre los comentarios por sa salida standard), si deseamos o no, que nos muestre la versión o el "Acerca de" la aplicación Mastín
También podremos cambiar las rutas que se muestran por defecto, tanto del ejecutable *java* como de los distintos paquetes necesarios para que se ejecute correctamante Mastín
Por último, si pulsamos el botón "Línea de comandos" situado en la parte inferior, nos aparecerá una ventana cuyo contenido será la línea de comandos equivalente al formulario que hemos rellenado en caso de que quisiésemos ejecutar Mastin en línea de comandos



Podemos seleccionar el texto, y copiarlo al portapaeles pulsando Control+C.
Podremos cerrar esta ventana pulsando el botón "Cerrar" situado en la parte inferior
Podemos ver el "Acerca de" de XMastin pulsando el botón "Acerca de" situado en la parte inferior de la ventana principal y nos aparece:



Pulsamos aceptar para cerrar dicha ventana
Si deseamos salir del programa XMastin, podemos pulsar en cualquier momento (excepto en el momento en que está trabajando) el botón "Cerrar" situado en la parte inferior derecha de la ventana principal.

Desinstalación

Usuarios de Windows

La desinstalación es incluso más sencilla que la instalación. Tan sólo hay que hacer click en "desinstalar Mastin v1.0 y XMastin v1.0" dentro de "Menú de inicio|Programas|XMastin v1.0"

Usuarios de Linux

La desinstalación es incluso más sencilla que la instalación. Tan sólo hay que hacer click en "desinstalar Mastin v1.0 y XMastin v1.0" dentro del escritorio