UNIVERSIDAD POLITÉCNICA DE CATALUÑA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN DE BARCELONA

PROYECTO FINAL DE CARRERA

Evaluación de la herramienta EJBCA para un Prestador de Servicios de Certificación

AUTOR:

Juan Manuel Alor Osorio

DIRECTOR:

Miguel Soriano Ibañez

Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona

“A la meva ávia Angelica, no vaig poder ser amb tú i ara m’esperas eternament. A totes les personas que m’han donat suport durant aquests dos anys”

INDICE

CAPÍTULO 1: INTRODUCCIÓN

Este proyecto final de carrera se enmarca dentro del área temática de la certificación digital, que forma parte del mundo de la criptografía asimétrica, concretamente de las infraestructuras de clave pública. Cada entidad de estas infraestructuras posee un par de claves criptográficas las cuales pueden estar almacenadas en *hard tokens* o en *soft tokens*. Dentro de la certificación digital, los prestadores de servicios de certificación son los encargados de emitir certificados digitales a través de sus autoridades de certificación, y estas autoridades realizan su actividad utilizando un software de certificación.

En este proyecto se pretende evaluar si la herramienta *opensource* EJBCA, la cual es un software de certificación, permite satisfacer las necesidades del prestador de servicios de certificación Firmaprofesional, S.A. De ahí que el título de este proyecto sea *"Evaluación de la herramienta EJBCA para un Prestador de Servicios de Certificación".*

Estas necesidades son originadas por el próximo cambio de software de certificación de la empresa prestadora de servicios de certificación, que pasará del que utiliza actualmente a EJBCA. Este cambio de software implica una migración de su autoridad de certificación principal, y esta migración, cinco necesidades principales, las cuales son:

[A] La determinación de una secuencia de pasos inequívoca para la instalación del nuevo software de certificación.

[B] El poder emitir con el nuevo software todos los tipos de certificados que el prestador emite.

[C] Que se realice una simulación del proceso de migración de la autoridad de certificación principal del prestador, que tiene sus claves criptográficas en un hard token HSM, al nuevo software de certificación.

[D] Que la autoridad antes importada reconozca como suyos, en el nuevo software, todos los certificados que emitió con el software antiguo.

[E] La integración del nuevo software a la arquitectura de certificación del prestador a través de algún método de acceso.

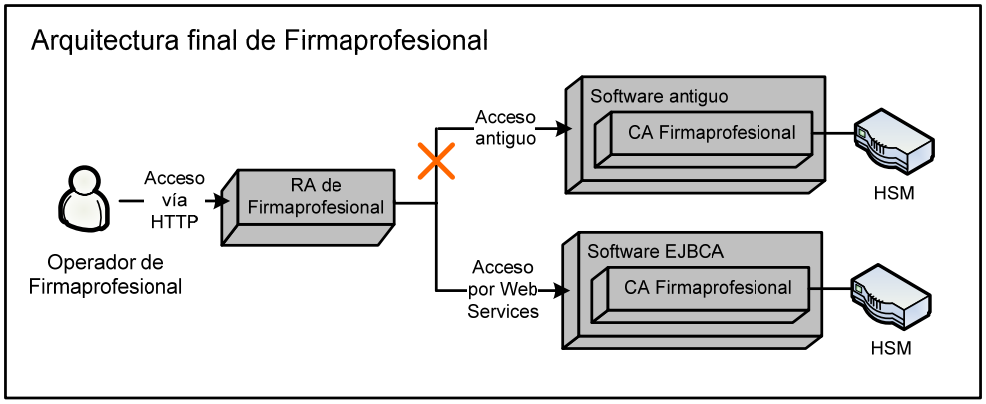
Sin embargo, no en todos los casos se pudo hacer exactamente lo especificado. Para resolver la necesidad del punto **[C]** se utilizó en la simulación una autoridad certificadora creada con el software EJBCA y con claves almacenadas en un *soft token* porque, por motivos de presupuesto, no se tuvo acceso a un *hard token* HSM. Esta diferencia supone un trabajo de investigación adicional del que se ocupará el prestador de servicios de certificación.

Por otro lado, la necesidad del punto **[E]** será resuelta íntegramente por el prestador, que utilizará el acceso por *Web Services* para este propósito. Para la del punto **[D]** se utilizaron unos certificados de prueba emitidos por la autoridad que se creó con EJBCA.

A lo largo de esta memoria se resolverán completamente cada una de las demás necesidades. Para esto primero se creó, en una primera instalación de EJBCA, una jerarquía de certificación similar a la del prestador y luego se importó esta jerarquía en

una nueva instalación del software. Este segundo paso constituye una simulación del proceso de migración real que realizará la empresa en un futuro.

En la siguiente figura se muestra como quedaría la arquitectura de certificación de Firmaprofesional luego de la migración. De ella se observa que en esta arquitectura los operadores acceden vía HTTP a la autoridad de registro (RA) y es esta autoridad la que se comunica con la autoridad de certificación (CA). El éxito total de la migración será que se cambie el software de certificación de la CA sin que este cambio sea sensible para el resto de componentes de la arquitectura.



1.1 Motivación

El 19 de diciembre del 2003 el congreso español aprobó la Ley 59/2003 que regula la firma electrónica en España, en esta ley se introduce la denominación: *"Firma electrónica reconocida"* que sirve para identificar a la firma digital que cumple con los requisitos necesarios para ser considerada equivalente a la manuscrita. También introduce la firma electrónica a las personas jurídicas no solo en el ámbito de gestión de tributos, como había sido hasta esa fecha, sino en todos los ámbitos telemáticos posibles.

Esta ley nace para generar confianza entre los consumidores hacia las transacciones electrónicas y uno de sus objetivos es potenciarlas. En este contexto los prestadores de servicios de certificación juegan un papel muy importante ya que son los responsables ante la ley de la correcta expedición de certificados digitales y de dispositivos seguros de creación de firma, elementos que generan una firma electrónica reconocida.

Firmaprofesional es uno de estos prestadores de servicios de certificación, centra su actividad en prestar servicios a todo tipo de empresas, colegios y corporaciones profesionales. A estos proporciona los instrumentos necesarios para que acrediten su identidad digital, y la de sus profesionales, sin riesgos a través de Internet, uno de estos instrumentos es la firma electrónica. Para lograr su propósito cuenta con una jerarquía de certificación dentro de la cual la autoridad certificadora es la encargada de generar y emitir los certificados digitales.

Actualmente, utiliza el software propietario KeyOne, que es propiedad de Safelayer, que está instalado en su autoridad certificadora para la emisión de los certificados digitales. Este software es fundamental en el desarrollo del negocio del prestador y a la vez le implica una alta inversión por el coste de la licencia. Sin embargo actualmente las alternativas opensource han alcanzado un alto nivel de desarrollo tanto así que equiparan o incluso superan las funcionalidades que puede ofrecer un determinado

software propietario, además de que la gran mayoría de ellas no tienen coste de licencia.

Ante esta nueva posibilidad Firmaprofesional se ha trazado el objetivo estratégico de migrar su actual jerarquía de certificación basada en un software privativo a una basada en software opensource. Una operación tan delicada como esta no se podía realizar sin una minuciosa evaluación previa de las funcionalidades que ofrecía la solución opensource escogida y de la viabilidad de esta migración.

En el presente trabajo se ha realizado una evaluación de la alternativa opensource EJBCA. En realidad las opciones no eran muy diversas, OpenCA en su momento fue una alternativa agradable pero la discontinuidad del proyecto hizo que se perdiera el interés por ella, NewPKI está basada en OpenSSL y desarrollada en C++ lo que hace que no sea independiente de la plataforma. Ante este panorama es de agradecer que exista una solución opensource tan completa como la que se ha escogido. EJBCA no es solo una autoridad certificadora, ofrece además las funcionalidades necesarias para construir a partir de ella una PKI seria, robusta, de alto rendimiento, flexible e independiente de la plataforma.

1.2 Objetivos

El objetivo principal de este proyecto es realizar una evaluación de la herramienta EJBCA para establecer si satisface los requerimientos propios de la actividad del prestador de servicios de certificación y según esto determinar la viabilidad de una migración.

La realización de este objetivo principal se logra alcanzando primero otros objetivos secundarios que se detallan a continuación, uno a uno éstos trazan la línea de trabajo que ha seguido la realización de este proyecto.

- Entender el sistema de producción de certificados de Firmaprofesional, esto es importante para determinar que es lo que queremos conseguir con EJBCA. Antes se tendrá que realizar un estudio de los conceptos teóricos que están englobados dentro del de *Infraestructura de Clave Pública*.

- Determinar una secuencia de pasos inequívoca para la instalación de EJBCA y de las herramientas adicionales que necesita. También, identificar los errores más comunes, y encontrar sus soluciones, que se pudieran presentar durante esta parte.

- Crear con EJBCA una jerarquía de certificación similar a la de Firmaprofesional, esto implica unas autoridades certificadoras que sean lo más parecidas posible a las que tiene Firmaprofesional.

- Crear en EJBCA los perfiles de certificación con los que se puedan emitir todos los tipos de certificado que emite actualmente Firmaprofesional, para esto habrá que analizar las prácticas y políticas de certificación de Firmaprofesional.

- Realizar una simulación del proceso de migración de la jerarquía de certificación creada a una nueva instalación de EJBCA. Esta simulación incluye la importación y exportación de las autoridades certificadoras y de los perfiles de certificación creados.

- Encontrar un procedimiento para importar a la jerarquía, de la nueva instalación de EJBCA, los certificados que se emitieron en la primera jerarquía creada.

1.3 Contenido de la memoria

Esta memoria se puede dividir en tres partes principales bien definidas, estas partes se resumen en los apartados siguientes. Luego de ellas se exponen las conclusiones finales del trabajo realizado seguidas de las líneas futuras de desarrollo del mismo. Se ha redactado también un glosario de términos en el que se incluyen las palabras que puedan sonar extrañas a los lectores menos conocedores del tema, la memoria concluye con la enumeración de las fuentes bibliográficas utilizadas.

1.3.1 Análisis

Actualmente, las infraestructuras de clave pública o PKIs son la base de los sistemas de seguridad cuya finalidad es mantener a salvo los datos más sensibles de una comunicación electrónica. Estas infraestructuras se basan en la identificación de entidades finales por medio de certificados digitales, los cuales son emitidos por autoridades certificadoras que garantizan la identidad de la entidad.

Dentro de una PKI cada entidad posee un par de claves con las cuales puede firmar electrónicamente documentos y encriptar los datos de una comunicación. Además, en una infraestructura se garantizan la confidencialidad, autenticación, integridad y no repudio de los elementos de la comunicación. De la combinación de los conceptos de PKI y del estándar de certificados digitales X.509 nace el concepto arquitectura PKIX, este concepto define los elementos clave de la arquitectura así como también las funciones de gestión que realizan. Dentro esta arquitectura la autoridad certificadora es el elemento más importante.

Los prestadores de servicios de certificación (PSC) son los encargados de desplegar infraestructuras de clave pública robustas que les permitan emitir certificados correctamente. Asimismo, luego de que se aprobara la ley de firma electrónica del 2003 son los responsables ante la ley de proporcionar a las empresas y a sus empleados mecanismos confiables para que cada persona, sea jurídica o no, tenga un firma electrónica equivalente a manuscrita. La empresa Firmaprofesional es uno de estos PSC, y está reconocido por el Ministerio de Industria como empresa acreditada para la emisión de certificados digitales válidos ya que cumple con los requisitos establecidos por la ley para este propósito.

Firmaprofesional cuenta con una arquitectura basada en una jerarquía de certificación que está regida por lo determinado en su declaración de prácticas y políticas de certificación. En esta declaración están definidas las características de los diferentes tipos de certificado y los servicios que ofrece la empresa. Para este fin utiliza un software de certificación de licencia de pago llamado KeyOne, este software está instalado en las autoridades certificadoras de la jerarquía las cuales son parte vital en la arquitectura de certificación de la empresa. El objetivo de la empresa es migrar de este software de pago a un software opensource llamado EJBCA.

EJBCA es una autoridad certificadora multifuncional, robusta, escalable, desarrollada en JAVA, de altas prestaciones y basada en componentes. Además, utiliza una serie de conceptos específicos que le permiten realizar sus principales funciones. Estas características junto con el que permita integrar solo algunos componentes de la herramienta a una arquitectura ya desplegada, el que permita administrar remotamente el componente CA utilizando el acceso por Web Services, el que sea flexible de configurar y el que ofrezca soporte para módulos HSM hacen de la herramienta la solución preferida para la migración.

También dispone de una interfaz de administración gráfica por medio de la cual se pueden diseñar los diferentes tipos de certificado y poner en marcha los servicios que el prestador ofrece a sus clientes. Era muy importante que luego de la migración se pudiera seguir realizando todo lo especificado en la declaración de prácticas y políticas de certificación, también que se reutilizaran las claves de la autoridad certificadora principal almacenadas en un módulo HSM y que la herramienta permitiera la importación de todos los certificados que la autoridad emitió anteriormente. Las funcionalidades que ofrece EJBCA satisfacen estos requerimientos.

1.3.2 Inicialización

Las partes que componen la inicialización de la herramienta son tres: el proceso de instalación de EJBCA, la configuración del sistema de registro de logs y la lista de los errores detectados durante la ejecución de las dos primeras partes.

Luego de algunos intentos se determinó la secuencia exacta de pasos que se ha de seguir para lograr una exitosa instalación de EJBCA. Estos pasos se explican como si se hicieran en un ordenador al que se le acaba de instalar el sistema operativo Linux distribución *Ubuntu 7.10*.

Los programas que necesita la herramienta son el *Java Development Kit* junto con las librerías *JCE Bouncy Castle* y *JCE Unlimited Strength Jurisdiction Policy Files*. La instalación más común de EJBCA, que es la que se lleva a cabo en este proyecto, es la que utiliza JBoss como servidor de aplicaciones y MySQL como gestor de base de datos. La instalación de MySQL necesita de las librerías del controlador JDBC para habilitar la conectividad con JAVA. Después de haber instalado los programas y librerías anteriores, y habiendo verificado que se tiene la utilidad *ant* instalada, se iniciará la instalación propia de EJBCA. Para esto antes hay que editar tres de los ficheros de configuración de la herramienta, uno de ellos define la configuración de la primera autoridad de certificación que se crea en la instalación, el otro define los parámetros web de las interfaces gráficas y el último los parámetros de acceso a la base de datos. Luego de modificarlos, se ejecutarán los comandos *ant* que hay que utilizar para completar la instalación del software, estos comandos son básicamente tres: *ant bootstrap*, *ant install* y *ant deploy*, y se deben utilizar siguiendo un orden específico. Finalmente, y para comprobar que la instalación ha sido exitosa, se ha de verificar que se pueda acceder correctamente a las interfaces gráficas admin web y public web de la herramienta.

La configuración de un sistema de registro de *logs* es una parte importante de la inicialización de la herramienta, para este propósito se configurará el paquete *Apache Log4j*. El llevar un registro de los mensajes de *logging* de EJBCA es importante para tener constancia en cualquier momento de los errores que se pudieran generar en la herramienta. Esta configuración se realiza en dos partes, primero se tendrá que configurar la herramienta modificando la sección correspondiente al almacenamiento de logs del fichero *ejbca.properties*, y luego el servidor de aplicaciones, añadiendo entradas *appender* y *category* al fichero *jboss-log4j.xml*.

También se ha incluido, como parte de la inicialización de la herramienta, una lista con los errores detectados durante el proceso de inicialización de EJBCA, a cada error se le ha registrado el tipo de error que es, su causa y la solución adoptada para solucionarlo. Se ha considerado importante registrarlos para que en un futuro, y en una eventual reinstalación, si se obtuvieran los mismos errores se conocieran sus causas y cómo resolverlos.

1.3.3 Configuración

La configuración de EJBCA es tal vez el capítulo más importante de este proyecto, debido a que trata la resolución de la mayoría de las necesidades del prestador de servicios de certificación. Se compone de tres partes: configuraciones previas, la configuración de la jerarquía de certificación del prestador y la simulación del proceso de migración.

Existen algunas funciones básicas en la herramienta que se deben conocer y configurar previamente antes de empezar a utilizarla. Entre éstas está la personalización de la interfaz admin web de acuerdo a las necesidades del administrador que la utilizará, también es importante conocer el proceso de diseño de certificados porque se empleará más adelante. La creación de publishers y de grupos de administradores son procedimientos interesantes de conocer pero menos relevantes.

La configuración en EJBCA de la Jerarquía de Certificación de Firmaprofesional es uno de los objetivos de este proyecto. Para realizarla, primero se crearon las autoridades certificadoras que utiliza la empresa, es decir la autoridad de certificación raíz y la subordinada, a esta última también se le llama autoridad principal porque es la encargada de emitir los certificados de entidad final. Luego de esto se crearon los perfiles de certificación necesarios para poder emitir certificados de entidad final que cumplan con las políticas de certificación de Firmaprofesional, durante esta creación se encontraron algunas limitaciones que se resolvieron personalizando la herramienta. Por último se crearon unas entidades finales de prueba, cada una con su respectivo certificado, que serán las que se utilizarán para simular la importación de certificados. Tanto las autoridades como las entidades finales se crearon con claves almacenadas en *soft tokens*.

El propósito de la creación de esta jerarquía es el poder luego simular la migración real del prestador de servicios de certificación, esta simulación se realizará exportando de la primera instalación las partes más importantes de la jerarquía para luego importarlas en una nueva instalación de EJBCA. Estas partes son la autoridad certificadora principal, los perfiles de certificación y las entidades finales que había emitido la autoridad.

En la simulación del proceso de migración lo primero que se tiene que importar es la autoridad certificadora principal, para esto antes hay que exportar sus claves criptográficas a un fichero PKCS12 que será el que se importará luego, para este propósito se utilizaron los comandos CLI *exportca* e *importca*. Lo segundo a importar en la nueva instalación son los perfiles de certificación creados anteriormente porque son los que permitirán emitir los certificados del prestador, para hacerlo se utilizaron los comandos CLI *exportprofiles* e *importprofiles*. Lo último que se tiene que importar son los certificados de prueba que se crearon anteriormente, esta importación se realiza con el comando CLI *importcert* y es la que demuestra que en la futura migración real será viable importar en EBCA todos los certificados que haya emitido la autoridad subordinada del prestador.

CAPÍTULO 2: ANÁLISIS

En este capítulo se realiza un análisis de los elementos más importantes que se utilizan o intervienen en el desarrollo de este proyecto. Primero se estudian los conocimientos teóricos necesarios para entender entorno de desarrollo de este proyecto, luego se presenta a la empresa prestadora de servicios de certificación Firmaprofesional como elemento interventor en este proyecto y se analiza su actividad, por último se realiza una descripción de la herramienta EJBCA y se analizan sus principales características.

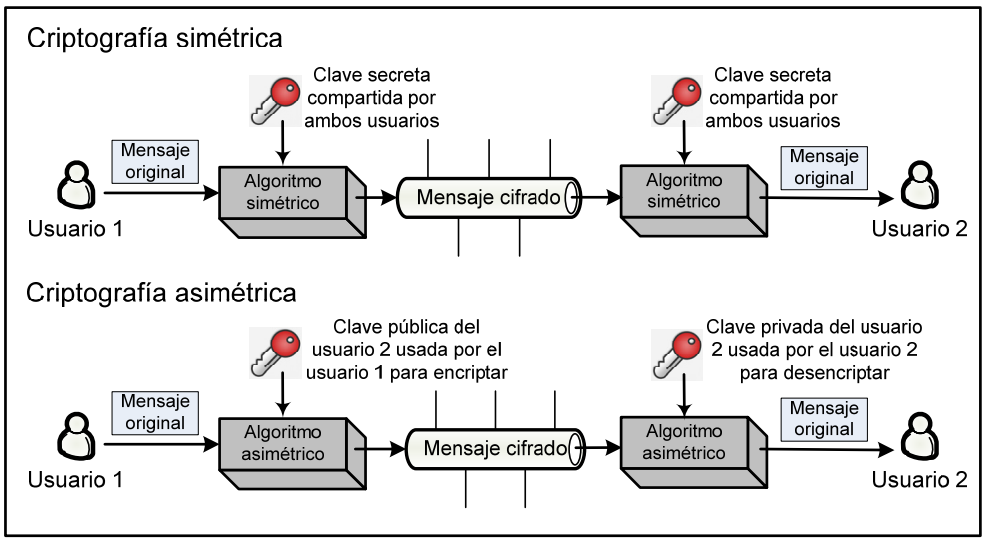
2.1 Conceptos Generales

En este apartado se realiza una breve introducción de los conceptos teóricos que engloba una infraestructura de clave pública, basada en la criptografía asimétrica, y de los elementos principales que forman parte de ella.

2.1.1 Infraestructura de Clave Pública (PKI)

La criptografía es el arte y ciencia de mantener los mensajes seguros, el criptoanálisis es el arte y ciencia de romper los textos cifrados, ambas ciencias se engloban en la rama de las matemáticas llamada criptología *(Schneier, Applied Cryptography, 1994, p.1).* Actualmente se desarrollan sistemas de seguridad que aplican la criptografía para mantener seguros los datos críticos de cualquier tipo de comunicación, como lo puede ser una transacción electrónica.

En el mundo de la criptografía moderna se pueden diferenciar dos claras vertientes, la criptografía simétrica y la criptografía asimétrica o de clave pública. La primera se basa en algoritmos simétricos que usan una misma clave para encriptar y desencriptar mensajes, y la segunda se basa en algoritmos de clave pública que usan claves distintas para la encriptación y desencriptación. Una infraestructura de clave pública (PKI, *Public Key Infrastructure*) es el conjunto de hardware, software, personas, políticas y procedimientos que se necesitan para crear, manejar, almacenar, distribuir y revocar certificados digitales basados en la criptografía asimétrica. Así está definida en el *RFC 2882 (Internet Security Glossary)*, a continuación se muestra una figura de estos dos tipos de criptografía.

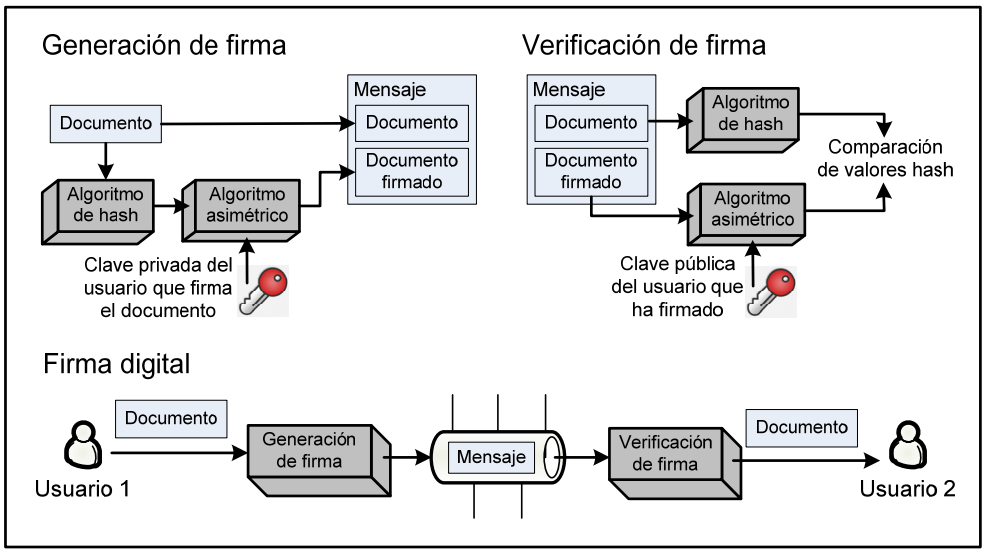


Según William Stallings *(Cryptography and Network Security, 2006, p.428)*, el objetivo de construir una PKI es permitir adquisiciones de claves públicas eficientes, seguras y convenientes. Cumplir este objetivo puede ser complicado, y no porque sea tecnológicamente complicado sino porque el problema reside en realizar una correcta implementación de los conceptos teóricos. Por esto es que actualmente empresas y grupos de trabajo especializados en temas de PKI se dedican a desarrollar e implementar el software necesario para construir infraestructuras de clave pública robustas y confiables. Empresas como IBM, Entrust, ChosenSecurity, RSA Security, Cyber Trust, Safelayer y Microsoft, entre otras, ofrecen soluciones PKI en forma de productos que los prestadores de servicios de certificación (PSCs) de cada ciudad pueden adquirir para montar su jerarquía de certificación.

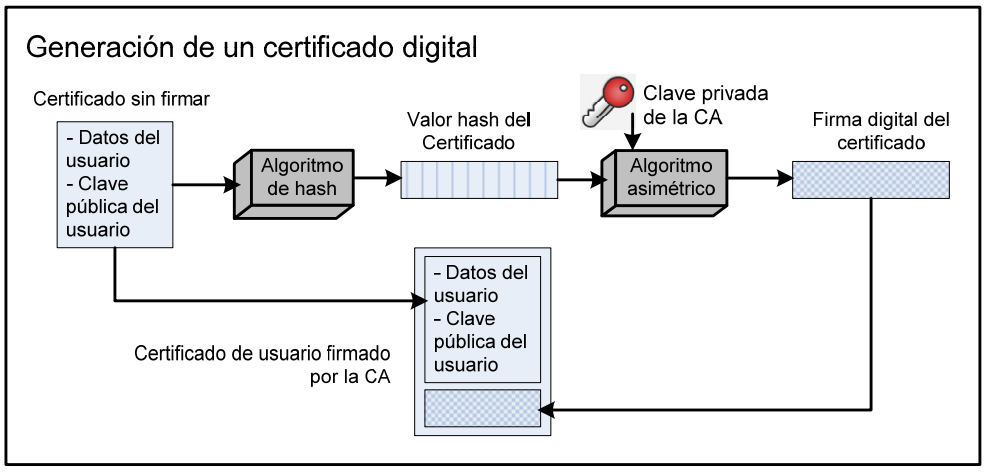
2.1.1.1 El certificado digital y la firma digital

En el proceso de firmado digital los algoritmos *hash* o *hash codes* tienen un papel importante, son criptosistemas de resumen que aceptan de entrada un mensaje de longitud abierta y tras aplicar un algoritmo devuelven una cadena de bits de longitud fija, esta longitud varía según el tipo de algoritmo que se use. Los algoritmos *hash* más conocidos actualmente son los *Message Digest* (siendo el más usado el Message Digest 5 ó MD5) y los *Secure Hash Algorithm* (siendo los más usados el SHA-1 y el SHA-256). Al valor obtenido luego de haber aplicado el algoritmo hash se le llama *valor del hash*, o simplemente *hash*, y tiene algunas características interesantes como que el valor que produce cada mensaje es único, que no puede haber otro mensaje que produzca el mismo valor y que es imposible reconstruir el mensaje a partir de su *hash*.

En el proceso de obtención de la firma digital de un mensaje, el involucrado posee dos pares de claves distintas que se llaman clave privada y clave pública, la clave privada se usa para encriptar y debe permanecer secreta mientras que la otra se usa para desencriptar y debe permanecer en un certificado digital. Además, se debe acordar en un algoritmo de firma y en un algoritmo hash que usarán tanto el firmante como la persona que verifique la firma. El primer paso es obtener el valor hash del mensaje y en el segundo paso el firmante debe utilizar su clave privada para encriptar este valor hash, a estos dos pasos se les conoce como firmar digitalmente un mensaje y al resultado obtenido como *firma digital*. La verificación de la firma digital implica que otra persona desencripte la firma digital utilizando la clave pública del firmante, si la puede desencriptar correctamente puede confiar en que el firmante realmente firmó el mensaje utilizando su clave privada. El resultado de la verificación es lo que se encriptó con la clave privada, en este caso el valor *hash* del mensaje. Si además el verificador tiene una copia del mensaje original en su poder, puede calcular el valor *hash* del mensaje y compararlo con lo obtenido en la desencriptación, si los valores son iguales entonces puede confiar en que el mensaje que tiene es el mismo que el que recibió firmado. En la siguiente figura se muestra el proceso de obtención de la firma digital explicado.



Un certificado digital es emitido por una autoridad certificadora (CA, *Certificate Authority*), y garantiza que esta CA ha verificado y confía en la identidad de la persona a la que pertenece el certificado. La CA manifiesta esta conformidad firmando el certificado y adjuntando la firma al final del mismo certificado, esta firma es efectuada con la clave privada de la CA que solo ella conoce. Esto permite que cualquier receptor del certificado digital pueda verificar, utilizando la clave pública de la CA, que el certificado ha sido firmado por ella. Otro documento que suele firmar la CA es la lista de revocación de certificados. En la siguiente figura se muestra el proceso de generación de un certificado digital.



El certificado asocia una única clave pública a una persona, esta clave se adjunta con el contenido del certificado junto con otros parámetros de la clave. En el proceso de firmado digital se utiliza el certificado digital para constatar que la clave pública, necesaria para la verificación de la firma, pertenece al firmante. El certificado contiene además un conjunto de datos relevantes que son definidos en la recomendación ITU-T X.509 por medio de una estructura de certificado, la estructura actual está en su versión 3 y también se le llama formato de certificado X.509 v3.

2.1.2 Características de una PKI

Las infraestructuras de clave pública tienen en cuenta cuatro aspectos fundamentales en cualquier comunicación electrónica, estos son especialmente importantes si se

habla de una transacción electrónica. A continuación se comentan brevemente cada uno de ellos y cómo la tecnología PKI los salvaguarda:

- **Confidencialidad**: Una comunicación entre dos personas no debe ser vista ni interferida por una tercera persona que no forme parte de la comunicación. La tecnología PKI usa la **encriptación** para asegurar la confidencialidad de los datos críticos que se encuentran en tránsito en la comunicación actual. También ofrece la posibilidad de encriptar datos valiosos almacenados en los servidores que tienen que conectarse a Internet, esto es importante porque si estos datos llegaran a ser robados el ladrón tendría que romper la encriptación antes de que pueda sacar ventaja de los mismos. Cabe resaltar que la confidencialidad también tiene un costo asociado, y es que las aplicaciones que utilicen la encriptación deben estar en servidores potentes que tengan los recursos computacionales necesarios para ofrecer una encriptación confiable y segura.

- **Autenticación**: Significa que el acceso a una comunicación electrónica debe estar restringido solo a quienes puedan presentar las credenciales necesarias de identidad. La forma más común de acreditar la identidad es a través de un identificador de usuario y una contraseña, esta forma está considerada como un bajo nivel de autenticación y es frecuentemente fácil de romper. La tecnología PKI utiliza el **certificado digital** como credencial de identificación. Una compañía puede especificar en que CAs confía y esto significaría que sólo acepta como válidos los certificados digitales emitidos por estas CAs. Un certificado digital puede ser guardado o en el explorador Web o en una tarjeta inteligente criptográfica (Cryptographic smart card). La segunda opción ofrece un nivel más de seguridad ya que el certificado digital nunca se copia al ordenador. Normalmente siempre se almacena encriptado y puede estar protegido con un código PIN o una contraseña.

- **Integridad**: Los datos recibidos deben ser los mismos que los datos enviados, esto quiere decir que no deben haber sido cambiados en el tránsito a su destino ya sea por error o a propósito. Otro aspecto que cuida la integridad es que se pueda probar innegablemente que dos copias diferentes del mismo documento son idénticas o no, tanto en el presente como en el futuro. La tecnología PKI utiliza los **algoritmos hash** para asegurar la integridad de estos datos. Las características de estos algoritmos hacen que cualquier cambio producido en el mensaje original durante su tránsito por la red se detecte como una pérdida de integridad. Normalmente se envía el mensaje y el valor de su hash, el receptor calcula el hash del mensaje recibido y lo compara con el valor del hash que le envió el transmisor, si los valores son idénticos se puede asegurar que el mensaje no ha sido modificado.

- **No-repudio**: Significa que si surge una discrepancia sobre lo que sucedió en una comunicación electrónica que implica intercambio de datos, habrá innegable evidencia presente dentro del sistema de comunicación que pueda ser utilizada para probar con suficiente certeza lo que realmente sucedió. Esto es especialmente sensible en operaciones que implican la firma de contratos electrónicos, en las cuales se evitaría que cualquiera de las partes que están implicadas en el contrato repudie lo que ha firmado. La tecnología PKI provee no-repudio a través del uso de la **firma digital**. Por ejemplo, el usuario A firma digitalmente el contrato y envía al usuario B el contrato firmado y el contrato en su versión original. Si el B puede verificar correctamente la firma de A también puede confiar en que A, y no otra persona, firmó el contrato. Si además B calcula el valor *hash* del contrato original y este valor es idéntico al que obtuvo

de la verificación, entonces puede confiar en que el contrato que tiene es el mismo que el que recibió firmado de A. B ahora con toda confianza podrá también firmar digitalmente el contrato y enviárselo a A. Así ambas partes habrán firmado el contrato y las pruebas de esto serán: el contrato original, el contrato firmado digitalmente por A y el contrato firmado digitalmente por B. Con estas pruebas no se podrá repudiar que en su momento A y B estuvieron de acuerdo y firmaron el mismo contrato, ni cambiar el contrato posteriormente ya que su valor hash no coincidiría con el que cada parte tiene. Sólo se podría alegar que la clave privada había sido robada al momento de firmar el documento, pero este problema se evita si además en el proceso de firmado se utilizan los sellos de tiempo o *timestamps*.

Existen otras tecnologías que salvaguardan algunos de estos aspectos utilizando algunos de los elementos que componen la tecnología PKI. Las *Virtual Private Networks* (VPNs) o el *Secure Socket layer* (SSL) utilizan la encriptación PKI para asegurar sólo la confidencialidad de los datos que fluyen a través de la red, y también hacen uso de los certificados digitales PKI para asegurar la autenticidad de la identidad de los elementos de la red. La tecnología PKI como conjunto puede ser usada como una solución segura y completa para las aplicaciones de intercambio de datos sensibles en Internet.

2.1.3 Infraestructura de Clave Pública X.509 (PKIX)

Como se ha visto, los certificados digitales son una parte fundamental de la tecnología PKI. Los esfuerzos por desarrollar una arquitectura basada en certificadas en Internet llevaron a adoptar el modelo de arquitectura basado en los certificados X.509 desarrollado por el grupo de trabajo PKIX del IETF. Actualmente el término PKIX se refiere a la infraestructura de clave pública basada en certificados X.509 (*Public Key Infrastructure* X.509) y el término certificado PKIX usualmente hace referencia a los perfiles de certificado y de listas de revocación basados en el estándar de certificados X.509v3. El grupo de trabajo PKIX ha producido una serie de estándares para satisfacer las necesidades de una PKIX, como el *RFC 3280 (Certificate and CRL Profile)*, el *RFC 2560 (Online Certificate Status Profile)*, el *RFC 3161 (Time Stamp Protocol)*, entre otros.

Los elementos clave que componen el modelo arquitectónico PKIX son:

- **Entidad final (End Entity)**: Es el nombre genérico que reciben los usuarios de una PKI o los dispositivos que forman parte de ella como ruteadores o servidores. Estos pueden ser identificados en el campo que define al propietario del certificado X.509. Las entidades usuarios normalmente utilizan los servicios que ofrece la PKI y los dispositivos normalmente los soportan.

- **Autoridad Certificadora (CA)**: Es la autoridad encargada de emitir los certificados digitales X.509 y usualmente también las listas de revocación de certificados (CRLs), aunque a veces delega la función de emitir CRLs a un elemento Emisor CRL. También puede desempeñar funciones administrativas como las de registro de entidades finales o publicación de certificados pero normalmente estas funciones son desempeñadas por las autoridades de registro. Existen como mínimo dos tipos de CAs en una jerarquía de certificación: la CA raíz (RootCA) y la CA subordinada (SubCA). La primera es la que emite certificados a otras CAs y la que cuyo certificado ha sido autofirmado. La segunda es la que emite certificados de entidad final y cuyo

certificado ha sido firmado digitalmente por la CA raíz. En una jerarquía pueden haber una o varias CAs subordinadas.

- **Autoridad de Registro (RA)**: Es un elemento opcional de la arquitectura PKIX que puede asumir algunas funciones administrativas de la CA, tal vez la más común sea el proceso de registro de las entidades finales, pero también puede realizar otras funciones como el proceso de revocación de certificados y el manejo de los datos de la entidad final. Dentro de una jerarquía de certificación pueden existir una o varias autoridades de registro, las cuales pueden ser internas o externas a la jerarquía de certificación del PSC.

- **CRL Issuer**: Es un elemento opcional de la arquitectura PKIX a el que la CA puede delegar la función de emitir las CRLs. Algunas veces se encuentra integrado en la CA a modo de servicio.

- **Repository**: Es el término que hace referencia a cualquier método existente para almacenar certificados y CRLs, y así poder ser obtenidos por las entidades finales. Uno de estos métodos es el protocolo LDAP.

En ciertas jerarquías de certificación puede existir un sexto elemento que se encarga de dar información sobre la vigencia de los certificados digitales. Este elemento se llama Autoridad de Validación (*Validation Authority*, VA), y recoge la información de qué certificados han sido revocados de las CRLs. Esta autoridad puede utilizar el protocolo OCSP para prestar los servicios de validación y no está incluida en la arquitectura PKIX porque es común que estos servicios los preste la misma CA. Pero en caso se quieran aislar los datos de la comprobación de la vigencia de un certificado de los datos de identidad de su titular se recomienda usar una autoridad de validación distinta de la de certificación.

En la arquitectura PKIX también se pueden definir unas funciones de gestión de la arquitectura, estas funciones son:

- **Registro**: Este es el proceso por el cual una entidad final (en concreto un usuario) registra sus datos directamente en una CA o por medio de una RA. También incluye procedimientos especiales de mutua autenticación, para los cuales se suelen generar claves privadas compartidas.

- **Inicialización**: Es el proceso por el cual el cliente se inicializa, de forma segura, con su clave pública y con otra información relevante de la CA en la que se ha registrado. La información de esta CA es la que se utilizará en el camino de validación de certificados Esto es necesario para que el sistema del cliente pueda operar correctamente.

- **Certificación**: Es el proceso por el cual la CA crea un certificado que certifica que una determinada clave pública pertenece a una entidad final, y se lo retorna al sistema del cliente o lo almacena en un repositorio.

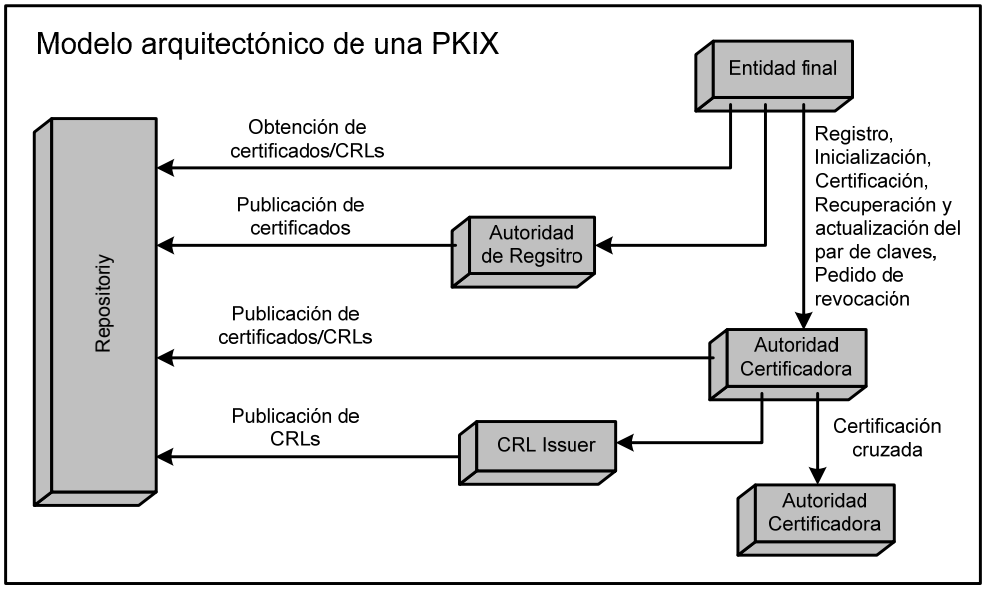
- **Recuperación del par de claves**: Este proceso permite a las entidades finales volver a obtener su par de claves, para esto las piden a una entidad autorizada de resguardo de claves. Usualmente la misma CA que emitió el certificado hace el papel de esta autoridad. Esta función es importante porque si se ha perdido el acceso a la clave de desencriptación no se podrán recuperar los datos encriptados.

- **Actualización del par de claves**: Es el proceso por el cual luego de un tiempo determinado se actualizan las claves de una entidad final y se le vuelve a emitir el respectivo certificado. Normalmente sucede cuando se cumple el tiempo de validez de un certificado o cuando el certificado ha sido revocado, y se realiza volviendo a crear un nuevo par de claves.

- **Pedido de revocación**: Ocurre cuando una persona autorizada avisa a la CA que ha ocurrido un suceso anormal y pide la revocación del certificado de una entidad final. Los motivos del pedido de revocación pueden ser o el compromiso de la clave privada o el cambio de nombre de la entidad, entre otros.

- **Certificación cruzada**: Es el proceso en el cual dos CAs intercambian la información necesaria para emitir un certificado cruzado. Este certificado es el que una CA raíz emite a una CA subordinada, siempre y cuando esta última CA cuente con una clave de firma autorizada para emitir certificados, normalmente, de entidad final.

A continuación se muestra una figura en la que se muestra la interacción entre los elementos del modelo PKIX.



Estas funciones necesitan ser soportadas por protocolos de gestión de PKIX, existen dos protocolos de gestión definidos por el grupo de trabajo PKIX. Uno de estos dos es el protocolo CMP (*Certificate Management Protocols*) que está definido en el *RFC 2510* y que tiene cada una de estas funciones implementadas.

En una PKI existe un documento importante llamado “Prácticas y Políticas de Certificación”, todo PSC debe tener publicado este documento para que esté disponible a cualquier usuario. En el *RFC 3647 (Certificate Policy and Certification Practices Framework)* se describen cada uno de los puntos que necesitan ser cubiertos por este documento. Entre otras cosas un PSC define en este documento las políticas de certificación que utiliza para la emisión de certificados digitales.

2.2 Firmaprofesional (FP)

Firmaprofesional S.A. nació en el 2001 como un prestador de servicios de certificación a raíz de un proyecto de algunos colegios profesionales. Con el propósito de que actúe como autoridad certificadora de los colegios profesionales y que a la vez sea totalmente independiente de ellos. Una de las motivaciones fue la idea de que la actividad de identificación de los profesionales de cada colegio no podía ser delegada a terceras personas por ser una actividad delicada y que no podía ser llevada a cabo por entidades externas que no conozcan su actividad.

Actualmente también presta servicios de certificación a empresas o instituciones tanto públicas como privadas, pero debido a sus orígenes cuenta con una especial sensibilidad y conocimiento directo de los colegios profesionales a los que facilita los instrumentos necesarios para su identificación electrónica.

En resumen tiene como principal actividad la generación, emisión y distribución de los certificados digitales y de los dispositivos y aplicaciones necesarios que acrediten la identidad de las empresas y la de sus profesionales en la red de redes. Con esto pretende proporcionar a las instituciones clientes los instrumentos necesarios para que puedan seguir ejerciendo a través de Internet las actividades que vienen realizando hasta ahora en el entorno tradicional. Y no solo a las instituciones sino también a los profesionales o empleados de cada una de ellas a los que facilita el ejercicio de su profesión y el establecimiento de relaciones profesionales por la red.

2.2.1 La Ley 59/2003

Firmaprofesional cumple con las disposiciones legislativas que dictan normativas en el ámbito de la certificación digital y en especial con las que dicta la Ley 59/2003 de firma electrónica, vigente a fecha de hoy. Además cumple con los requisitos impuestos por esta ley para que sea acreditado como un prestador de servicios de certificación certificado. El organismo que verifica que cumpla con estos requisitos es el Ministerio de Industria, Turismo y Comercio, en el cual se encuentra registrado.

En el enlace https://www11.mityc.es/prestadores/busquedaPrestadores.jsp se puede verificar esta acreditación buscando al prestador Firmaprofesional, S.A. Además, esta característica le permite expedir certificados digitales, también llamados certificados electrónicos, reconocidos como válidos.

En su exposición de motivos la ley recalca que *"Los datos más recientes señalan que aún existe desconfianza por parte de los intervinientes en las transacciones telemáticas…constituyendo esta falta de confianza un freno para el desarrollo de la sociedad de la información, en particular, la Administración y el comercio electrónicos"*. Ante esta problemática identifica como solución, entre otras alternativas, a la firma electrónica y la define como *"un instrumento capaz de permitir una comprobación de la procedencia y de la integridad de los mensajes intercambiados a través de redes de telecomunicaciones, ofreciendo las bases para evitar el repudio, si se adoptan las medidas oportunas basándose en fechas electrónicas."*. Luego define a los sujetos autorizados para expedir los certificados electrónicos que hacen posible el uso de la firma electrónica cuando dice que *"Los sujetos que hacen posible el empleo de la firma electrónica son los denominados prestadores de servicios de certificación. Para ello expiden certificados electrónicos, que son documentos electrónicos que relacionan las herramientas de firma electrónica en poder de cada usuario con su identidad personal".*

Hasta antes de la promulgación de esta ley sólo se permitía a las personas jurídicas utilizar los certificados electrónicos en su interacción con la administración pública, como podían ser los trámites de gestión de tributos. Con esta ley la firma digital de una persona jurídica podrá ser utilizada en múltiples aplicaciones, especialmente en procesos automatizados como la realización de pedidos o la emisión de facturas electrónicas. Firmaprofesional ofrece certificados electrónicos a las personas jurídicas que quieran acogerse a los beneficios que les ofrece esta ley, agilizando así la vida de las sociedades.

Esta ley también introduce el término de firma electrónica reconocida, con el que se refiere a la firma digital avanzada que está basada en un certificado reconocido y que ha sido creada por un dispositivo seguro de creación de firma electrónica. Si cumple estos requisitos la firma electrónica será vista como equivalente a la firma manuscrita, y el empleo de la misma tendrá las mismas implicaciones legales que la firma manuscrita. Se le llama dispositivo seguro de creación de firma (DSCF) a cualquier programa o sistema informático que garantice que las claves criptográficas puedan ser producidas solo una vez, que estas no pueden ser derivadas de la propia firma, que la firma este protegida contra la falsificación y que no altere los datos a firmar.

Un certificado reconocido es un certificado electrónico expedido por un prestador de servicios de certificación que cumple con los requisitos de comprobación de identidad y otros datos específicos del solicitante. Estos certificados están basados en el formato que especifica el certificado de clave pública X.509, además tienen que incluir, al menos, los siguientes datos:

- El código identificativo único del certificado. Especificado en el campo **número de serie** de un certificado X.509.

- El comienzo y el fin del período de validez del certificado. Especificado en el campo **período de validez** de un certificado X.509.

- La identificación del prestador de servicios de certificación que emite el certificado. Especificado en el campo **nombre del emisor** de un certificado X.509

- La identificación del firmante, en el caso de personas físicas, generalmente por su nombre, apellidos y su número de documento nacional de identidad o a través de un seudónimo que conste como tal de manera inequívoca. En el caso de personas jurídicas, por su denominación o razón social y su código de identificación fiscal. Especificado en el campo **nombre del sujeto** de un certificado X.509.

- Los datos de verificación de firma que correspondan a los datos de creación de firma que se encuentren bajo el control del firmante. Especificado en el campo **información de la clave pública del sujeto** y en la extensión X.509v3 **identificador de la clave del sujeto** de un certificado X.509.

- Los límites de uso del certificado, si se establecen. Especificado en la extensión X.509v3 **uso de la clave** de un certificado X.509.

- La indicación de que se trata de un certificado electrónico reconocido. Especificado en la extensión X.509v3 **políticas de certificación** de un certificado X.509.

- Los límites del valor de las transacciones para las que puede utilizarse el certificado, si se establecen. Especificado en la extensión **valor límite del certificado cualificado (QC Limit value)** de un certificado X.509.

- La firma electrónica avanzada del prestador de servicios de certificación que expide el certificado. Especificado en el campo **firma digital** de un certificado X.509.

Los certificados reconocidos que emite Firmaprofesional cumplen con lo expuesto en la ley, lo que quiere decir que incluyen estos datos.

Otra consideración importante que hace esta ley es sobre el futuro DNI electrónico, al cual en el Título II, Capítulo III define como *"el documento nacional de identidad que acredita electrónicamente la identidad personal de su titular y permite la firma electrónica de documentos"*, y reconoce su eficacia para *"acreditar la identidad y los demás datos personales del titular que consten en el mismo, y para acreditar la identidad del firmante y la integridad de los documentos firmados…"*. También sienta las bases jurídicas que marcarán su regulación, así como los requisitos y características del documento. La norma prevé que con el DNI electrónico los instrumentos utilizados en el proceso de firma electrónica se volverán más conocidos, utilizados y comercializados. Firmaprofesional tiene desplegada la infraestructura tecnológica necesaria para ofrecer los servicios adicionales que se necesiten cuando el DNI electrónico sea una realidad.

2.2.2 Uso de los certificados emitidos

Aunque la única acreditación necesaria para poder emitir certificados electrónicos reconocidos es la otorgada por el Ministerio de Industria, Turismo y Comercio, los certificados que emite Firmaprofesional también están homologados por la Agencia Catalana de Certificación (CATCert), por el Ministerio de Administraciones Públicas (MAP), por la Agencia Tributaria (AET) y por muchos otros organismos más. Estas homologaciones son voluntarias y sirven para que los certificados que emite la empresa sean aceptados como válidos por esos organismos.

Los certificados que emite Firmaprofesional pueden tener numerosos usos y aplicaciones. Además, éstos se incrementarán cuando el DNI electrónico sea una realidad ya que este documento permitirá a los ciudadanos identificarse de la misma forma que con el documento nacional de identidad tradicional en los diferentes trámites que tenga que realizar a través de la red. También permitirá a los ciudadanos que lo posean, firmar cualquier clase de documentos y estar seguros de la integridad del documento firmado digitalmente luego de haber realizado la firma.

En términos generales los usos se pueden agrupar en dos categorías:

- **Firma electrónica de documentos:** Como la firma de correos electrónicos, firma de facturas, de recetas, de expedientes, de proyectos, de cuentas financieras, de visado y de otros documentos.

- **Autenticación Web:** Para garantizar la identidad de un usuario ante terceros dentro del ámbito telemático. Como pueden ser el pago de impuestos, la presentación de declaraciones juradas, el registro de documentos electrónicos, consulta de declaraciones, pago de aduanas e impuestos especiales, entre otros trámites permitidos por la Seguridad Social, el Ministerio de Fomento, el Ministerio de Sanidad y Consumo y otras instituciones más.

2.2.3 Las Prácticas y Políticas de Certificación

Los prestadores de servicios de certificación están obligados a efectuar una supervisión y gestión permanente de los certificados electrónicos que expiden, los procedimientos y prácticas que desarrollen a fin de cumplir con esta gestión deben estar recogidas en un documento llamado “Declaración de prácticas de certificación” (en inglés *CPS o Certification Practice Statement*). Además este documento debe estar disponible al público de manera fácilmente accesible, como mínimo de forma electrónica y de forma gratuita.

En este documento se especifican las condiciones aplicables a la solicitud, expedición, uso, suspensión y extinción de la vigencia de los certificados electrónicos. Además, se explican las obligaciones del prestador en cuanto a la gestión de los datos de creación de firma y de certificados electrónicos. También se habla de las medidas de seguridad técnicas y organizativas de la empresa, así como de los tipos de perfiles que existen y de los servicios de validación de certificados disponibles.

El incurrir en alguna de las infracciones señaladas en la ley de firma electrónica expondría a Firmaprofesional a sanciones de entre 30.000 y 600.000 euros según la misma ley. Una infracción puede ser el incumplir alguna de sus obligaciones, y entre ellas están la protección de los datos personales del usuario, el no almacenar los datos de creación de firma, el proporcionar al usuario de forma gratuita las prácticas de certificación, el tener que contar con un seguro de responsabilidad de al menos 3.000.000 de euros y el comunicar a la autoridad competente y a todos los usuarios de un próximo cese de actividad si se diera el caso. Un ejemplo de infracción grave es guardar una copia de la clave criptográfica privada de la persona a la que se le haya prestado servicio.

Adicionalmente a las condiciones generales establecidas en la CPS, cada tipo de certificado emitido por Firmaprofesional se rige por unas condiciones particulares de emisión recogidas en un documento denominado *“Política de Certificación”* (en inglés *CP o Certificate Policy).* Existe una política de certificación por cada tipo de certificado emitido.

Tanto las CPS como las CPs así como otras políticas que haya emitido Firmaprofesional están identificadas por un único OID (*Object Identifier*). Para esto la empresa tiene registrado en el registro oficial de parámetros de Internet IANA el número “13177” como OID de empresa privada, por lo tanto todos los OIDs que le pertenecen empiezan por “1.3.6.1.4.1.1377” y el significado de cada uno de ellos se muestra en el siguiente cuadro:

OID	Tipo de Objeto	Descripción
10.0.V.R	Declaración de Prácticas de Certificación (CPS)	V = Versión de la CPS R = Subversión de la CPS
10.1.T.D	Políticas de Certificación	T = Tipo de Certificado 1 =Colegiado 2 = Vinculada 3 = Servidor Seguro SSL 4 = Servicio Seguro (VA/TSA) 5 = Persona Jurídica 6 = Factura Electrónica 7 = Firma de Código personal 8 = Firma de Código corporativo 9 = Firma Móvil D = Dispositivo 1 = DSCF 2 = Software
10.1.5.1.X	Extensión del Certificado de Persona Jurídica	1 = Mancomunado 2 = Datos Registrales
10.10.1	Política de Certificación de Autoridad de Certificación Subordinada	CA Subordinada
20.0.1	Política de Sellado de Tiempo	TSA Firmaprofesional

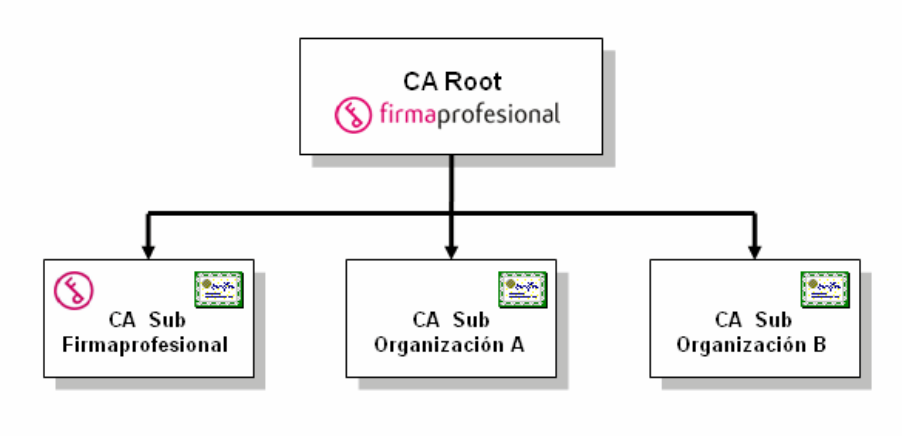
Las políticas de certificación de Firmaprofesional se clasifican en dos grupos: las que definen certificados reconocidos por la ley 59/2003 y las que no. Asimismo el primer grupo se divide entre las políticas que definen certificados que usan *Dispositivos Seguros de Creación de Firma (DSCF)* y las que no. En el siguiente cuadro se muestra cada política de certificación junto con sus OIDs:

CERTIFICADOS RECONOCIDOS SEGÚN LEY 59/2003	
En Dispositivo Seguro de Creación de Firma:	
1.3.6.1.4.1.13177.10.1.1.1	CP de Colegiado
1.3.6.1.4.1.13177.10.1.2.1	CP de Persona Vinculada
1.3.6.1.4.1.13177.10.1.5.1	CP de Persona Jurídica
1.3.6.1.4.1.13177.10.1.6.1	CP de Factura Electrónica
1.3.6.1.4.1.13177.10.1.9.1	CP de Firma Móvil
En Software:	
1.3.6.1.4.1.13177.10.1.1.2	CP de Colegiado
1.3.6.1.4.1.13177.10.1.2.2	CP de Persona Vinculada
1.3.6.1.4.1.13177.10.1.5.2	CP de Persona Jurídica
1.3.6.1.4.1.13177.10.1.6.2	CP de Factura Electrónica
OTROS CERTIFICADOS NO RECONOCIDOS:	
1.3.6.1.4.1.13177.10.1.3.1	CP de Servidor Seguro (SSL)
1.3.6.1.4.1.13177.10.1.4.1	CP de Servicio Seguro (TSA/VA)
1.3.6.1.4.1.13177.10.1.7.1	CP de Firma de Código personal
1.3.6.1.4.1.13177.10.1.8.1	CP de Firma de Código corporativo

En adelante, este trabajo se ocupará solo de las políticas de los certificados reconocidos por ley, ya que son las más delicadas debido a que su uso está regulado por una entidad pública. En el anexo A se ofrece más detalle de las políticas de certificación de Firmaprofesional y de los certificados que definen.

2.2.4 La Jerarquía de Certificación Firmaprofesional

Firmaprofesional tiene su propia jerarquía de certificación digital llamada "*Jerarquía de Certificación Firmaprofesional*". Está basada en la existencia, en el nivel superior, de una CA raíz y en los niveles inferiores, de una serie de CAs subordinadas vinculadas a la misma. Adicionalmente existen una serie de elementos que proporcionan servicio al proceso de certificación asociado de una o varias de las CAs subordinadas constituidas. Se aprecia mejor dicha jerarquía en una figura.



Entre los elementos adicionales que participan en el proceso de certificación tenemos a la *"Autoridad de homologación de Políticas*" o PA, que se encarga de la administración de las prácticas y políticas de certificación y está compuesta por los miembros del departamento de operaciones de Firmaprofesional. Luego está la *"Entidad Registro*", que es la entidad que tiene constancia de la lista de de certificados emitidos y su estado, esta entidad es la que se encarga de verificar como válidos los documentos y datos que proporciona la entidad suscriptora antes de que se le emita el certificado. Las funciones que realiza esta entidad le han sido delegadas por las CAs.

La CA raíz de Firmaprofesional se llama *"Autoridad de Certificación Firmaprofesional CIF A62634068"* y la CA subordinada principal, encargada de emitir los certificados de entidad final, tiene de nombre *"AC Firmaprofesional - CA1"*. La CA raíz publica CRLs cada seis meses o en el momento en que el certificado de una de sus CAs subordinadas deja de ser válido. La CA subordinada principal, la CA1, las emite cada 24 horas o en el momento en que el certificado de una de sus entidades finales deja de ser válido.

Las *"Autoridades de Registro*" o ARs llevan a cabo principalmente las funciones de registro delegadas por la CA1 y pueden ser internas o externas. Tanto los colegios profesionales como las personas jurídicas cliente pueden actuar como ARs externas a la jerarquía de Firmaprofesional, permitiéndoles así participar en el proceso de certificación lo cual supone una garantía adicional para ellos. Si la AR es interna la propia Firmaprofesional S.A. actúa como tal.

El marco de funcionamiento, las obligaciones y las responsabilidades de cada entidad de la jerarquía están declarados en las prácticas y políticas de certificación. Existen algunas entidades adicionales que ofrecen los servicios de sellado de tiempo, como la "Autoridad de Sellado de Tiempo", ésta última se rige por su propia política particular no incluida en la declaración anterior.

2.2.5 Los servicios que ofrece

Firmaprofesional emite diferentes tipos de certificados, las características de cada tipo de certificado vienen descritas en su correspondiente "Política de Certificación". Algunos estos certificados son reconocidos para firma electrónica y otros son destinados a usos distintos de la firma electrónica reconocida.

Certificados de persona física reconocidos según la Ley 59/2003 de firma electrónica:

- **Certificados de Colegiado**: Son certificados reconocidos de persona física que identifican al suscriptor como profesional colegiado por un Colegio Oficial.

- **Certificados de Persona Vinculada**: Son certificados reconocidos de persona física que identifican al suscriptor como vinculado a una determinada organización, ya sea como empleado, asociado, colaborador, cliente o proveedor.

- **Certificado de Factura Electrónica**: Son certificados reconocidos de persona física con poderes de representación de una organización para firmar facturas electrónicas.

- **Certificado de Firma Móvil:** Son certificados reconocidos de persona física destinados a servicios de firma móvil que cumplan con los estándares ETSI M-COMM MSS (*Mobile Commerce - Mobile Signature Service*), utilizando la tarjeta SIM del teléfono móvil como DSCF.

Entre los certificados de persona jurídica reconocidos por la misma ley están:

- **Certificados de Persona Jurídica**: Son certificados reconocidos de persona jurídica según el Art.7 de la Ley 59/2003.

Las claves privadas asociadas a estos certificados, menos el de Firma Móvil que solo se almacena en DCSF, pueden almacenarse en software o en Dispositivos Seguros de Creación de de Firma (DSCF).

Firmaprofesional también emite certificados digitales para usos diferentes a la firma electrónica reconocida. Estos certificados no son reconocidos según la Ley 59/2003 y son específicos a la necesidad del cliente:

- **Certificado de Servidor Seguro**: Son certificados utilizados para autenticar la identidad de un servidor mediante el uso de protocolos de comunicación cifrada como SSL, TLS o IPSEC.

- **Certificado de Servicio Seguro**: Son certificados que permiten firmar evidencias digitales, pueden ser emitidos por una Autoridad de Sellado de Tiempo (TSA) o por una Autoridad de Validación (VA). Su emisión y utilización requerirá las máximas garantías de seguridad.

- **Certificado de Firma de Código**: Son certificados utilizados para firmar código ejecutable, como por ejemplo un *applet* de Java, garantizando la autoría y su integridad frente a modificaciones no autorizadas. Este certificado puede ser personal o corporativo.

Adicionalmente, la empresa podrá emitir certificados no reconocidos de Persona Física y Jurídica, para entornos cerrados de usuarios (como certificados de operador o

administrador). Las características de estos certificados no se encuentran dentro de su CPS.

Firmaprofesional como PSC están obligado a mantener accesible al menos un servicio gratuito de validación del estado de vigencia de los certificados que ha emitido, en este servicio debe indicarse de manera actualizada si estos están vigentes, o si su vigencia ha sido suspendida o ha terminado. Este dato se obtiene examinando el campo del certificado que expresa entre que fechas es válido el certificado y también verificando que el número de serie del certificado no se encuentre en la última CRL emitida por la CA.

La empresa ofrece dos servicios de validación de certificados, el primero mediante consulta a las CRLs emitidas y el segundo mediante el servicio OCSP. El servicio OCSP está montado en un servidor con un certificado válido de VA.

También ofrece el servicio de sellado de tiempo, y lo hace incorporando una Autoridad de Sellado de Tiempo externa (*Time Stamping Authority*, TSA) en la estructura organizativa del cliente. Desvinculando así dicha autoridad del proceso de emisión del certificado, de forma que sea más flexible y ajustada a las necesidades del cliente. Esta autoridad se encarga de garantizar que cierto dato, como la firma digital, existió o fue creado en un instante determinado del tiempo. Los servicios de no-repudio requieren de esta habilidad para ser más robustos. Normalmente esta autoridad está desplegada en un servidor con un certificado válido de TSA.

La TSA puede ser también interna a Firmaprofesional, es decir que puede ser una autoridad más dentro de su jerarquía, en este caso el certificado de sellado de tiempo colgaría de dicha jerarquía.

2.2.6 Situación actual

Actualmente, Firmaprofesional utiliza en su jerarquía de certificación una solución PKI desarrollada por la empresa Safelayer llamada *KeyOne*. Esta es una familia de soluciones para sistemas de gestión de certificados digitales y para servicios avanzados de validación de certificados y sellado de tiempo. Concretamente utiliza el software *KeyOne CA* en sus CAs principales: CA raíz y CA1. Este software ofrece las funciones requeridas para emitir certificados digitales X.509v3 y además componentes opcionales que proveen funciones adicionales. Pero su licencia es de pago, por lo que Firmaprofesional realiza una inversión sustancial en conceptos de coste de licencia.

Las soluciones opensource están a la orden del día y se muestran como alternativas interesantes. Generalmente nacen como proyectos encabezados por escuelas o grupos de investigación de prestigio, lo cual implica una garantía de calidad del producto final. Las hay casi para todos los ámbitos tecnológicos y las infraestructuras de clave pública no son la excepción. Generalmente suponen libertad para modificar y distribuir el código fuente del software, pero no siempre suponen un coste de licencia cero. Casos como los de NewPKI, OpenCA y EJBCA son ejemplos de soluciones PKI sin coste de licencia.

Ante este abanico de posibilidades Firmaprofesional se ha planteado el objetivo estratégico de migrar su actual jerarquía de certificación, basada en *KeyOne*, a una basada en la solución PKI opensource EJBCA. Esta migración no es tan sencilla ya que debe aprovechar la mayor cantidad de componentes hardware de la arquitectura de certificación actual, conservar los pares de claves de sus CAs, lograr que el nuevo

sistema ofrezca los mismos servicios que el antiguo y reconocer como suyos los certificados emitidos por el sistema anterior. El presente trabajo realiza una evaluación del proceso de configuración de EJBCA y de la complejidad del proceso de migración.

2.2.7 Desarrollo de este trabajo

La información en la web relativa a EJBCA no es despreciable. Aún así, quedan temas de especial interés sin profundizar, como por ejemplo el proceso de implementación de EJBCA en diferentes escenarios a los habituales, la personalización de los perfiles de certificados, la importación de una CA y la de sus certificados emitidos.

La utilidad de este trabajo radica en que trata temas que se podrían englobar dentro de la experiencia de cambiar el software de certificación actual de un PSC a EJBCA. Así este proyecto contribuye a engordar aún más la documentación relacionada con la herramienta.

El desarrollo de este proyecto ha constado de diferentes partes, las cuales se fueron esclareciendo aún más mientras se avanzaba en el desarrollo. Las partes previas al desarrollo del proyecto en sí han incluido el estudio de temas de PKI y algunos algoritmos de encriptación, la utilización de PGP y OpenSSL, la familiarización con las herramientas de administración de un entorno Linux, el manejo de la tecnología JAVA y del servidor de aplicaciones JBoss. Las partes correspondientes al desarrollo del proyecto en sí incluyeron la determinación del proceso de instalación de la herramienta y de los paquetes que necesitaba, la creación de la *Jerarquía de Certificación de Firmaprofesional* y la simulación del proceso de migración de esta jerarquía.

A parte de estas, hay dos partes del proceso de migración real que no se trataron en este trabajo porque de ellas se encargó el departamento técnico de la empresa, estas son la integración por medio de *Web Services* de la RA de Firmaprofesional y el componente CA de EJBCA, y la importación de la CA subordinada principal de la empresa (de claves almacenadas en un HSM) a una instalación de EJBCA.

Firmaprofesional ha tenido una participación activa en la realización de este trabajo. En la dirección y supervisión del avance del proyecto, en el asesoramiento técnico y en el compartir la información con la que disponían. Todo esto con el fin de lograr completar el objetivo final del proyecto para el beneficio de ambas partes.

El entorno hardware en el que se desarrolló el trabajo lo componen una PC de pruebas locales y un servidor de pruebas remoto. Entre el software utilizado están EJBCA v3.5.2, JDK 1.5.0, las librerías JCE BouncyCastle v1.38 y Unlimited Strength Jurisdiction Policy v5.0, JBoss v4.2.2, MySQL v5.0 (cliente y servidor), MySQL Query Browser, MySQL Query Administrator, el driver JDBC v5.1.5, Ant 1.7.0, OpenLDAP v2.49, LDAP Browser/Editor v2.8.2, entre otras menos importantes.

2.3 EJBCA

EJBCA, en su página web oficial, es definida como una autoridad certificadora multifuncional, que basada en la tecnología J2EE, constituye un CA robusta, escalable, de alto rendimiento y basada en componentes. Se dice que es multifuncional porque puede realizar todas las funciones propias de una CA, y alguna más, sin delegarlas a otras herramientas. Se dice que está basada en componentes porque cada una de las funciones las realiza un componente de la misma herramienta.

Como autoridad certificadora que es su principal función es la emisión de certificados. Estos se pueden emitir para distintos propósitos como autenticación robusta de usuarios y dispositivos, comunicación segura con clientes y servidores SSL, firma y encriptación de correo electrónico, firma de documentos, acceso a sistemas usando tarjetas criptográficas, asegurar conexiones VPN y muchos otros. EJBCA, para realizar su actividad correctamente en un entorno de producción, necesita de un servidor de aplicaciones y una base de datos.

2.3.1 El proyecto EJBCA

El proyecto EJBCA fue iniciado por Tomas Gustavsson y Philip Vendil en el año 2001, en Noviembre de ese mismo año apareció la primera versión de EJBCA, la 1.0. Desde entonces se han publicado más de 50 nuevas versiones, actualmente, la última versión estable es la 3.7.1. Tiene un equipo de 15 desarrolladores, y está alojado en el repositorio de proyectos opensource más grande del mundo: SourceForge, en donde cuenta con gran actividad.

En este sitio mantiene dos listas de correos y cuatro foros públicos de discusión bastante utilizados donde se puede encontrar rápida respuesta a cualquier duda específica. En sí, cuenta con gran cantidad de información disponible en su página web oficial, en su sitio wiki, en su blog y en los mismos paquetes de instalación. EJBCA como solución PKI ha crecido y se ha popularizado bastante en los últimos años, el sitio wiki y el blog son nuevas fuentes de información que al inicio de este trabajo no existían. De aquí en adelante la aparición de nuevas ayudas e información sobre el tema se generará sola al mismo ritmo de aparición de nuevos usuarios.

2.3.2 Características y funcionalidades

EJBCA tiene tres funcionalidades principales: la de CA, la de RA y la de CRL Issuer. A parte de estas, ofrece funcionalidades adicionales como las de respondedor OCSP, cliente OCSP, almacenador y publicador de certificados y CRLs. También ofrece administración por línea de comando, notificaciones a usuarios por correo electrónico, informes del sistema, gestión y firma de logs, integración con un HSM, recuperación de claves, soporte para claves ECDSA, servicios que soportan los protocolos CMP, XKMS y SCEP, entre otros servicios.

Una funcionalidad importante que tiene la herramienta es la administración remota por Web Services, para esto tiene una interfaz llamada WSCLI, y a través de ella se puede acceder a su componente CA desde una RA externa. Esta funcionalidad es la que utilizará Firmaprofesional para integrar la herramienta a su arquitectura de certificación.

Dos de las características principales de la herramienta son su flexibilidad y su independencia de la plataforma, la primera gracias a que su arquitectura está basada en componentes y la segunda como consecuencia de estar desarrollada totalmente en JAVA. EJBCA puede utilizarse tanto para operar aislada, ya que no tiene dependencias con otras herramientas, o integrada en cualquier aplicación J2EE utilizando los componentes de la herramienta que se consideren necesarios.

2.3.2.1 Interfaces de la herramienta

Se tienen tres interfaces básicas de interacción con la herramienta: la *public web*, la *admin web* y la *command line interface* (CLI). La primera es una interfaz gráfica pública que es accesible por todos los usuarios en general y les permite obtener sus certificados, las CRLs de la CA y verificar la validez de cualquier certificado. La segunda es también una interfaz gráfica pero que es solo accesible por los administradores de la herramienta, a través de esta interfaz ellos pueden realizar tareas de administración. La tercera interfaz es una línea de comandos que permite realizar algunas operaciones de administración y es especialmente útil cuando se necesita realizar alguna acción por medio de scripts.

2.3.2.2 Interfaz de administración flexible

EJBCA ofrece un fácil y sencillo control de los privilegios de administración de los administradores de la arquitectura de certificación ya que es posible establecer grupos de administradores de CA, administradores de RA, superadministradores y supervisores. Por ejemplo a través de la admin web se puede nombrar como administradores de RA a los miembros de una unidad organizativa específica del PSC. Esto aumenta el nivel de seguridad de la infraestructura ya que las funciones de administración están diferenciadas por el tipo de administrador que se sea. Esta interfaz gráfica también permite controlar el formato de los certificados gracias a la definición de perfiles de certificado y de entidad final.

2.3.2.3 Unidad de firma

EJBCA permite elegir el tipo de unidad de firma que más convenga. Según la política de certificación que se quiera implementar se puede usar un módulo software para almacenar las claves de firma o se puede requerir el uso de HSMs o de tarjetas criptográficas.

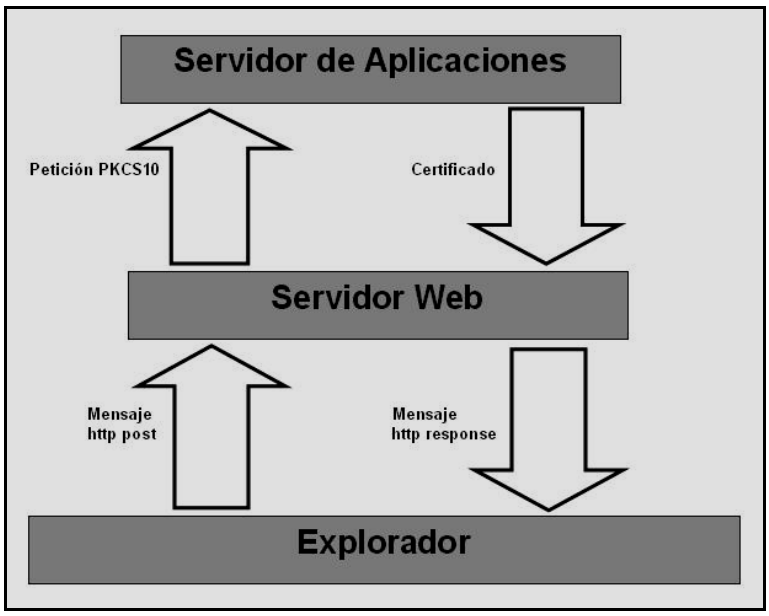
2.3.2.4 Almacenamiento de certificados

Los certificados que emite EJBCA se almacenan en la base de datos a la que está ligada, y a través de unos componentes publicadores llamados *publishers* se publican en directorios específicos. Esta funcionalidad es completamente configurable lo que permite que cada PSC pueda decidir cuándo y cómo sus certificados serán publicados, incluso se puede reemplazar el mecanismo de publicación por uno específico del PSC.

2.3.2.5 Obtención de los certificados de usuario

Luego de que se ha dado de alta un usuario se le tiene que emitir el certificado, antes en el proceso de registro se le ha debido crear un nombre de usuario y una contraseña. Con estos datos el usuario desde su explorador accede a la public web y se autentifica, crea su par de claves y envía la clave pública junto con sus datos de registro a la CA en formato PKCS10 (*Certification Request Standard*). Luego de esto la CA le crea el certificado al usuario y se lo envía. El mecanismo de autentificación descrito en este proceso se puede reemplazar por otro que se adecue a los requerimientos de seguridad del PSC.

En la figura se muestra un diagrama más detallado del proceso de obtención de certificados. El usuario desde el explorador interactúa con el servidor web, que es el que contiene los servlets de la aplicación, luego éste interactúa con el servidor de aplicaciones que contiene a la aplicación EJBCA y a sus componentes CA y RA.

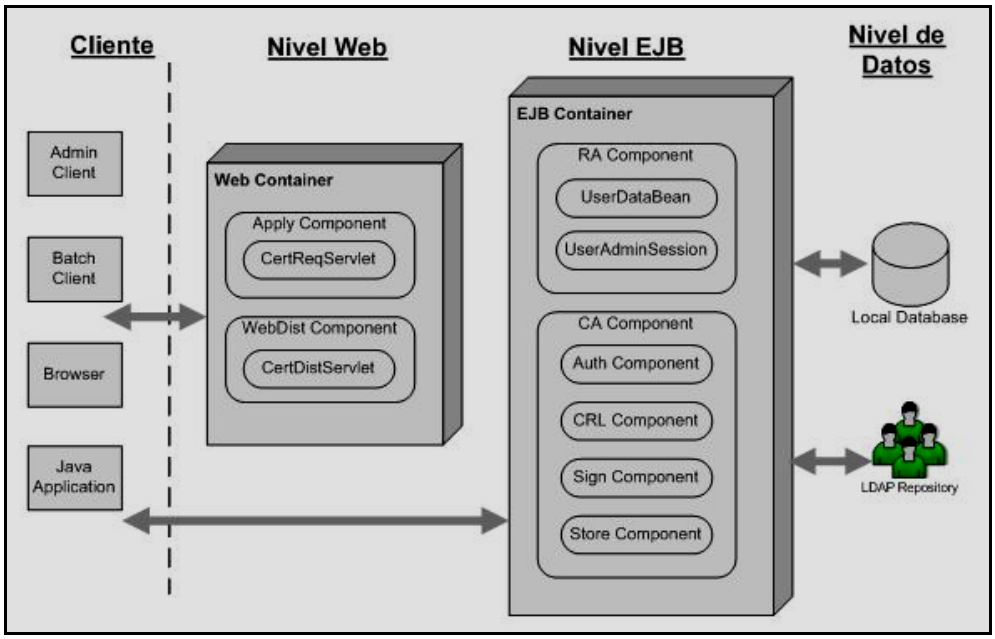


2.3.3 Arquitectura de la herramienta

EJBCA se ha programado usando la tecnología J2EE 1.3 que permite desarrollar software de arquitectura de N niveles, y que también incluye la especificación EJB 2.0 que es un API que proporciona un modelo de componentes distribuido en el lado del servidor. El uso de los *Enterprise Java Beans* (EJB) hace que EJBCA tenga una arquitectura basada en componentes, estos componentes son de por sí flexibles y sobre todo reutilizables.

La mejor forma de implementar una PKI es usando una arquitectura escalonada y basada en componentes que se pueda adaptar a las distintas necesidades y requerimientos únicos que tenga un PSC. EJBCA utiliza esta arquitectura y está formada por distintos componentes, estos pueden personalizarse o incluso reemplazarse a la hora de desplegar la herramienta editando las opciones de configuración. Esta personalización o reemplazo se realiza según la necesidad de cada PSC, como en el caso del almacenamiento y obtención de certificados.

En la figura se muestra un diagrama muy simple de la arquitectura de EJBCA, se observan tres niveles principales y cómo cada nivel está compuesto por distintos componentes. Los niveles Web y EJB son representados por sus respectivos contenedores. El contenedor web está compuesto por componentes formados por servlets y en una implementación normal es quien interactúa con la parte cliente. El contenedor EJB está compuesto por dos componentes principales: RA y CA, el componente RA se comunica con la base de datos a través de unos beans y el componente CA se comunica con el directorio repositorio a través de un componente especial.



EJBCA puede utilizar sus servlets y los EJBs o solo utilizar los EJBs. Los servlets son solo el *front-end* hacia los EJBs de cara al cliente y si la CA se despliega integrada a una aplicación J2EE se puede reemplazar este *front-end* por el propio de la aplicación. Por ejemplo en la figura se ve que el nivel EJB puede comunicarse directamente con una aplicación JAVA específica del cliente sin necesidad de usar los servlets.

2.3.4 Conceptos específicos de EJBCA

Esta herramienta utiliza algunos conceptos que son importantes para entender su funcionamiento, estos son:

- **Perfil de Certificado (Certificate Profile)**: EJBCA permite configurar el conjunto de características generales de un certificado en un perfil de certificado. Por ejemplo definen qué extensiones X.509v3 constarán en el certificado y si serán críticas o no. Algunas extensiones requieren un valor específico, en este caso todos los certificados emitidos con este perfil tendrán el mismo valor, como el caso de la extensión CRLDistributionPoint. Otras sólo requieren que se determine si deben estar presente o no, y si lo están, definir si el valor se especificará por perfil de entidad final o por entidad final, como en el caso de la extensión SubjectAlternativeName. En este perfil también se determina qué *publisher* utilizará, en caso de que los certificados deban ser publicados.

- **Perfil de entidad final (End Entity Profile)**: Permite configurar qué datos pueden o deben estar presentes a la hora de dar de alta a una entidad final, básicamente definen el nombre distintivo (en inglés *Distinguished Name* o *DN)* de la entidad. En este perfil se define a qué perfil de certificado estará ligado, y en conjunto los dos perfiles *diseñan* un certificado.

- **Publishers**: Un publisher es el encargado de almacenar los certificados en una ubicación central, se podría decir que un publisher almacena los certificados en el repositorio correspondiente. EJBCA ofrece publishers con soporte para LDAP.

- **Hard Token Profiles**: Este perfil contiene información como la longitud de clave que se usará en las tarjetas criptográficas, los perfiles de certificado que se podrán usar, y la plantilla de gráficos que se podrá imprimir en las tarjetas.

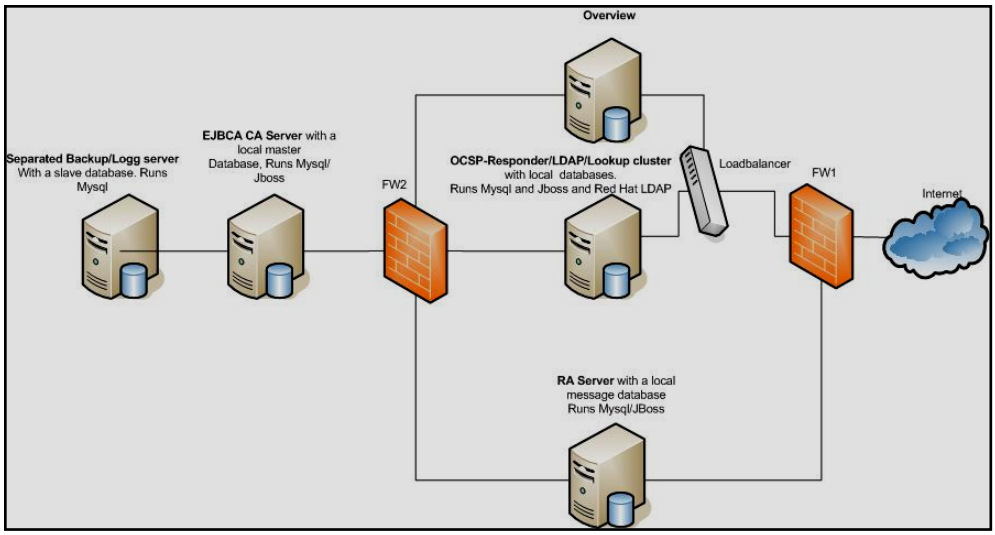
- **Hard Token Issuers**: Representa el lugar físico en donde se gestiona la emisión de tokens hardware, por ejemplo de tarjetas criptográficas.

- **External RA**: Se refiere al paquete extra que ofrece EJBCA para montar una RA externa a ella. Este paquete llamado *extra* es útil cuando se le quiere negar todo el tráfico entrante a la CA para dejar que ella misma coja la información que debe procesar de la RA.

- **OCSP responder**: Es un componente que se encarga de responder las consultas OCSP, puede estar en la misma CA o puede ser externo. Este componente envía una respuesta firmada con el certificado de la CA o con un certificado OCSP dedicado, según sea interno o externo.

2.3.5 Un ejemplo de implementación

En la figura se muestra un ejemplo de implementación de EJBCA, con él se pretende mostrar la flexibilidad de la configuración de la herramienta. Se ve que la herramienta se ha instalado en un servidor que hace solo de CA y está conectada a un servidor de backup en el que almacenan los logs y al que copia periódicamente toda su base de datos. Se tiene una RA externa que recibe las peticiones de certificado y un OCSP responder externo que recibe las peticiones OCSP, evitando de esta manera que la CA reciba tráfico directamente. Se tiene también un *firewall* que está configurado para que solo permita el tráfico saliente de la CA y no el tráfico entrante, lo que hará la CA será coger periódicamente las peticiones que necesite procesar tanto de la RA como del OCSP responder. Otro aspecto a resaltar es que se ha implementado un cluster de *OCSP responders* externos, esto es útil cuando se espera recibir muchas peticiones OCSP pero emitir pocos certificados por lo que no es necesario implementar un cluster de CAs. El balanceador de carga recibe el tráfico de Internet y lo redirige al *OCSP responder* que tenga menos carga en ese momento.



2.3.6 El porqué de su elección

Firmaprofesional eligió EJBCA como la solución PKI opensource a la cual migrará por la flexibilidad de configuración de sus componentes y por la facilidad con la que estos se pueden integrar dentro de su arquitectura de certificación. Por ejemplo, Firmaprofesional utiliza un mecanismo de entrega de certificados distinto al que utiliza EJBCA pero como este mecanismo es reemplazable no significa un inconveniente. Además ya cuenta un módulo RA desplegado por lo que en un futuro se podrían plantear solo integrar el componente CA de la herramienta, esta integración podría realizarse utilizando la interfaz web service o el servicio XKMS que ofrece EJBCA.

La herramienta es lo suficientemente robusta como para soportar infraestructuras de múltiples niveles de CAs dentro de una sola instancia de ejecución y se despliega correctamente dentro de ambientes de *clustering* para el que ofrece servicios de monitorización. Además está diseñada para soportar diferentes módulos HSM, entre los cuales están los nCipher, y dispositivos DSCF para el firmado con tarjetas criptográficas inteligentes. Firmaprofesional utiliza estas dos características, especialmente los módulos HSM nCipher para custodiar las claves privadas de sus CAs, por eso el que EJBCA tenga soporte para estos módulos fue una razón más para elegirla ya que facilitará la migración de claves de las CAs.

Otro motivo importante es que la herramienta utiliza la licencia LGPL (*Lesser General Public License*), la cual no tiene coste. La diferencia de esta licencia con la GPL es que puede ser integrada casi sin ninguna limitación con cualquier programa propietario. Algunos programas que también la usan son el navegador Mozilla y el proyecto OppenOffice.

La herramienta puede ser instalada en cualquier servidor de aplicaciones J2EE., aunque la instalación más sencilla y más probada es sobre JBoss también existe documentación de cómo hacerlo sobre otros servidores como GlassFish, Weblogic, OC4J o Websphere. EJBCA es independiente de la base de datos que se utilice ya que se encuentran en niveles distintos de la arquitectura, o sea se puede instalar tanto con Hypersonic, MySQL, PostgreSQL, Oracle, DB2, MS-SQL, Derby y Sysbase como con cualquier otra.

Firmaprofesional tiene montada su CA sobre un servidor linux y utiliza en su arquitectura de certificación los programas JBoss y MySQL, por eso el que EJBCA los soporte y el que sea independiente de la plataforma fueron razones de peso para su elección.

Entre otras características favorables de EJBCA está el que no solo soporta claves de firma RSA sino también las ECDSA. Esto es importante porque en un futuro, al ritmo con el que se desarrollan los ordenadores, las bases del algoritmo RSA fallarán y será posible romperlo. Ante esto, los algoritmos que se basan en la criptografía de curva elíptica son los que se perfilan como los algoritmos base de la criptografía de clave asimétrica del futuro, ECDSA es uno de ellos.

El que existan PSCs importantes que han instalado EJBCA en sus autoridades de certificación es una garantía adicional. Algunas de las instalaciones de referencia de EJBCA más importantes son las de la *Autoritat de Certificació de la Comunitat Valenciana*, con más de 200.000 certificados personales emitidos; el Grupo Safa, que se dedica a la industria de la salud en España; las del Ministerio de Defensa y Ministerio de Finanzas Francés, dos de los organismos públicos más grandes de Europa; y la del Cuerpo Nacional de Policía Sueco, que tiene emitidos ya más de

25.000 certificados de empleado y ofrece soluciones de firma para el pasaporte electrónico sueco.

CAPÍTULO 3: INICIALIZACIÓN

En este capítulo se desarrolla el procedimiento seguido para la instalación de la herramienta y de los componentes que necesita para que funcione correctamente, también se aborda la configuración del paquete de registro de *logs* que se utilizó para la gestión de los mismos, y por último, se listan los diferentes errores encontrados en las repetitivas instalaciones que se hicieron acompañados de las soluciones que los resolvieron. La determinación de una secuencia de pasos inequívoca para lograr una correcta instalación del software de certificación EJBCA es el mayor resultado de esta parte y será aprovechado tanto en el siguiente capítulo como en la futura migración de Firmaprofesional.

3.1 Proceso de instalación

En este punto se describirá el proceso de instalación de EJBCA, desde la creación de nuevas variables de entorno, de usuarios y de repositorios de datos, la instalación de ciertos componentes y herramientas necesarias hasta la modificación de ficheros específicos, estos pasos son necesarios para lograr un correcto funcionamiento de la herramienta EJBCA (versión 3.5.2). La instalación que se detalla a continuación se realizó en el servidor remoto *ovh1.firmaprofesional.com*, que tiene el sistema operativo GNU/Linux de distribución *Ubuntu Gutsy Gibbon 7.10*.

Antes de nada es recomendable ejecutar en una consola y como usuario privilegiado los siguientes comandos para verificar que se tienen instalados algunos programas útiles:

```
apt-get install autoconf lynx zip unzip tofrodos ldap-utils
apt-get install db4.2-util libldap2-dev libssl-dev libnet-ldap-perl
apt-get install libapache2-mod-jk tdsodbc sqsh junit libapr1 slapd
```

3.1.1 Instalación de Java JDK 1.5.0

Dado que EJBCA está basada en la tecnología J2EE es necesario instalar el JDK (*Java Devolopment Kit*) respectivo, en este caso se utilizó la versión 1.5.0 porque con esta versión se habían realizado pruebas anteriores.

El primer paso es obtener el paquete directamente de la página de Sun, *http://java.sun.com/downloads/*, y luego instalarlo, o se puede utilizar el comando *apt-get install.*

```
apt-get install sun-java5-jdk
```

Lo normal es que luego de esto el JDK se haya instalado correctamente pero puede darse el caso de que al ejecutar este comando se obtenga un error como el siguiente:

```
E: No se pudo encontrar el paquete sun-java5-jdk
```

Esto se debe a que la descarga de paquetes del repositorio "*multiverse*" no está habilitada en el fichero */etc/apt/sources.list*, para habilitarla se le debe descomentar la siguiente línea:

```
#deb-src http://es.archive.ubuntu.com/ubuntu gutsy universe
```

Y al final de la misma, añadir la palabra *multiverse*. Debería quedar de la siguiente manera:

```
deb-src http://es.archive.ubuntu.com/ubuntu gutsy universe multiverse
```

Luego de haber realizado correctamente la instalación del JDK y de haber resuelto las posibles dependencias que hayan podido aparecer, *apt-get install* resuelve los problemas de dependencias por sí mismo, y en caso de tener instalada otra versión de Java se debe establecer que versión del JRE se quiere utilizar. Por ejemplo, Ubuntu 7.10 tiene la versión 6 instalada por defecto y esta versión usa el paquete *java-6-sun*. El sistema operativo permite tener más de una versión instalada pero solo utilizar una en un determinado momento, en este caso se quiere que el sistema utilice la versión 1.5.0 (*java-1.5.0-sun*), esto se hace con el comando *update-java-alternatives*.

```
update-java-alternatives -s java-1.5.0-sun
```

El siguiente paso es añadir al fichero */etc/environment* la variable de entorno *JAVA_HOME*, para esto se le debe añadir la línea siguiente:

```
JAVA_HOME="/usr/lib/jvm/java-1.5.0-sun"
```

De ahora en adelante se hará referencia a la ubicación del JRE 1.5.0 utilizando *java_home*.

3.1.1.1 Las librerías Bouncy Castle

La versión 1.4 de Java introdujo el manejo de criptografía y basó la implementación de las clases que la proporcionan en una arquitectura llamada "*Proveedores de Seguridad*", paro lo que añadió dos extensiones: la JCA (*Java Cryptography Architecture*) y la JCE (*Java Cryptography Extension*). La primera proporciona la arquitectura para realizar las operaciones criptográficas y la segunda extiende las funcionalidades de la JCA añadiendo proveedores de seguridad. El proveedor de seguridad por defecto de Java es uno proporcionado por Sun, el cual no soporta demasiados algoritmos criptográficos debido a las restricciones legales establecidas por algunos países respecto a la exportación e importación de criptografía.

BouncyCastle es un proveedor de seguridad desarrollado por el grupo de programadores *Legion of BouncyCastle*, y es el utilizado por EJBCA ya que es de gran calidad, completamente gratuito y su licencia permite utilizarlo en cualquier tipo de aplicación.

Un ejemplo de un problema que se puede tener si se usa solo el proveedor de seguridad proporcionado por Sun es que no se podrán manipular los ficheros del tipo PKCS12 de EJBCA con el *keytool* de Sun. El estándar PKCS12 define un formato para almacenar claves privadas acompañadas de sus respectivos certificados digitales y

normalmente el *keytool* de Sun puede solo leer los ficheros PKCS12 pero no escribir sobre ellos, este problema lo soluciona la instalación de las librerías *BouncyCastle*.

Es importante resaltar que EJBCA no es compatible con otro JCE ya que solo *BouncyCastle* contiene clases que permiten no solo el uso de certificados sino también la generación de los mismos.

Para instalar las *BouncyCastle JCE* se tiene que descargar de la página *http://www.bouncycastle.org/latest_releases.html* la última versión de las librerías *bcprov-jdk15-XXX.jar* y *bcmail-jdk15-XXX.jar*, llamadas "*BouncyCastle provider"* y *"BouncyCastle SMIME/CMS"* respectivamente, y copiarlas en *java_home /jre/lib/ext/*que es donde se guardan las extensiones. Cuando se realizó esta instalación la última versión era la 138 (*XXX=138*).

Por último se debe modificar el fichero *java_home/jre/lib/security/java.security*, aumentando una línea en la definición de los *security providers* (proveedores de seguridad). Esta parte del fichero tiene esta apariencia:

```
#
# List of providers and their preference orders (see above):
#
security.provider.1=sun.security.provider.Sun
security.provider.2=sun.security.rsa.SunRsaSign
security.provider.3=com.sun.net.ssl.internal.ssl.Provider
security.provider.4=com.sun.crypto.provider.SunJCE
security.provider.5=sun.security.jgss.SunProvider
security.provider.6=com.sun.security.sasl.Provider
```

La línea a aumentar define al nuevo proveedor *BouncyCastle* y lo coloca en el segundo orden de preferencia, esta línea es:

```
security.provider.2=org.bouncycastle.jce.provider.BouncyCastleProvider
```

El fichero finalmente debe quedar de la siguiente manera:

```
#
# List of providers and their preference orders (see above):
#
security.provider.1=sun.security.provider.Sun
security.provider.2=org.bouncycastle.jce.provider.BouncyCastleProvider
security.provider.3=sun.security.rsa.SunRsaSign
security.provider.4=com.sun.net.ssl.internal.ssl.Provider
security.provider.5=com.sun.crypto.provider.SunJCE
security.provider.6=sun.security.jgss.SunProvider
security.provider.7=com.sun.security.sasl.Provider
```

Es importante que el primer proveedor de seguridad sea Sun y que el segundo sea BouncyCastle.

3.1.1.2 Las JCE Unlimited Strength Jurisdiction Policy Files

EJBCA es una de las aplicaciones reconocidas como exentas de las restricciones criptográficas existentes en algunos países, debido a que utiliza *"strong criptography"* y *"keystore passwords"* de longitud mayor a 7 caracteres necesita tener instalado el JCE

Unlimited Strength Jurisdiction Policy Files para funcionar correctamente. Este JCE se puede descargar de la página *http://java.sun.com/javase/downloads/index_jdk5.jsp*.

Se deberá descargar el correspondiente a la versión 1.5.0 de Java, el *Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 5.0*. El fichero que se obtendrá será el *jce_policy-1_5_0.zip.*

Al descomprimirlo se obtendrán los siguientes ficheros:

```
unzip jce_policy-1_5_0.zip
       creating: jce/
       inflating: jce/COPYRIGHT.html
       inflating: jce/README.txt
       inflating: jce/US_export_policy.jar
       inflating: jce/local_policy.jar
```

Finalmente, de estos ficheros se tendrán que copiar los ficheros JAR (*local_policy.jar* y *US_export_policy.jar*) a la carpeta *java_home/jre/lib/security/*, los otros ficheros se pueden borrar tranquilamente.

3.1.2 Instalación del servidor de aplicaciones JBoss

EJBCA necesita de un servidor de aplicaciones para ejecutarse, aparte la instalación por defecto y más sencilla de la herramienta está hecha sobre el servidor de aplicaciones JBoss, y es el que se usará. Es recomendable descargar la última versión estable de la página *http://labs.jboss.com/jbossas/downloads/*.

En este caso la última versión estable fue la 4.2.2. El fichero a instalar está disponible en los formatos ZIP y TAR-GZ, por comodidad se escogió el de formato ZIP llamado *jboss-4.2.2.GA.zip*. Se recomienda descomprimir este fichero directamente a */usr/local/* que es el directorio en el que se guarda el software que se desea conservar en una actualización del sistema. Esto se puede hacer fácilmente con el siguiente comando:

```
unzip jboss-4.2.2.GA.zip –d /usr/local/
```

Luego, se tendrá que añadir al fichero */etc/environment* la variable de entorno *JBOSS_HOME* que define la ubicación exacta de JBoss, esto es importante porque EJBCA utiliza esta variable para saber en donde se encuentra el servidor de aplicaciones. La línea a añadir es la siguiente:

```
JBOSS_HOME="/usr/local/jboss-4.2.2.GA/"
```

Una vez realizado lo anterior JBoss estará listo para usarse, los ficheros que controlan la ejecución del servidor se encuentran en el directorio */usr/local/jboss-4.2.2.GA/bin/*, se utilizan de la siguiente manera:

```
Para iniciar el servidor de aplicaciones JBoss
./usr/local/jboss-4.2.2.GA/bin/run.sh

Para pararlo (también se puede utilizar CTRL+C)
./usr/local/jboss-4.2.2.GA/bin/shutdown.sh -S
```

De ahora en adelante se hará referencia a la ubicación de JBoss 4.2.2 utilizando *jboss_home*.

3.1.3 Instalación de MySQL

El servidor de aplicaciones JBoss tiene su propia base de datos *in-mermory* llamada *"Hypersonic database"*, que sea *in-mermory"* significa que mientras más se utilice ocupará más memoria lo cual es un problema si se piensan emitir muchos certificados. Otro problema de esta base de datos es que no soporta todos los comandos SQL. Por estos motivos si se piensa construir un entorno de producción es recomendable instalar otra base de datos.

En este caso, como se está construyendo un entorno de producción, se utilizará el gestor de base de datos relacional MySQL. Para este propósito se necesitan instalar los paquetes *mysql-server* y *mysql-client*, los cuales a su vez utilizan los paquetes *mysql-server-5.0* y *mysql-client-5.0* que son los binarios de los primeros.

Si se quiere se puede instalar también una interfaz gráfica para manejar con más comodidad las bases de datos que se creen, una alternativa es utilizar MySQL *Administrator* conjuntamente con *MySQL Query Browser,* la primera servirá para administrar la base de datos y la segunda para ejecutar sentencias SQL. Para esto se necesitan instalar los paquetes *mysql-admin* y *mysql-query-browser*, los cuales a su vez dependen de los paquetes *mysql-server-common*, *mysql-client-common* y *mysql_common*.

La instalación de *mysql-server* creará por defecto un usuario *root* para el que pedirá que se ingrese una contraseña. Es recomendable que con este usuario se cree otro usuario que tenga asignados todos los privilegios y que sea el que realice todas las tareas de administración necesarias relativas a EJBCA. Para esto habrá que iniciar sesión en MySQL con el usuario *root* y usar el comando *GRANT*, al usuario creado se le llamó *ejbca* y se le identificó con la contraseña *ejbca*, a continuación se detallan los pasos:

```
Se inicia sesión como root
mysql -u root -p

Se crea el usuario ejbca identificado con contraseña ejbca
mysql> GRANT ALL ON *.* TO ejbca IDENTIFIED BY 'ejbca' WITH GRANT OPTION;
mysql> exit;
```

En la configuración por defecto de MySQL, la herramienta escucha las peticiones en el puerto *3306* y en la dirección *0.0.0.0*, que representa a todas las interfaces. Esta configuración se puede ver y modificar en el fichero */etc/mysql/my.cnf*.

Antes de proseguir con la instalación se deberá crear una base de datos en MySQL, en ella se crearán todas las tablas y, dentro de ellas, los datos que se generen al instalar y utilizar EJBCA posteriormente. A la base de datos creada se le llamó *ejbca_3_5_2*, a continuación se detallan los pasos:

```
Se inicia sesión como ejbca
mysql -u ejbca -p

Se crea la base de datos ejbca_3_5_2
mysql> create database ejbca_3_5_2
mysql> exit;

Se inicia sesión como ejbca directamente en la tabla creada
mysql -u ejbca -p ejbca_3_5_2
```

3.1.3.1 El driver JDBC

Para que MySQL funcione correctamente con Java será necesario instalar un controlador JDBC (*Java Database Connectivity*) el cual es un API que permite la conectividad entre el lenguaje JAVA y cualquier base de datos SQL existente. En este caso se usará el *Connector/J* que es el controlador JDBC oficial para MySQL. Este controlador es el que provee la conectividad con las aplicaciones desarrolladas en Java, EJBCA es una de ellas.

Lo primero será descargar dicho controlador de *www.mysql.com*, la versión del *Connector/J* disponible al momento de realizar esta instalación fue la 5.1, en todo caso se recomienda descargar la última versión. Se puede descargar esta versión directamente desde *http://dev.mysql.com/downloads/connector/j/5.1.html*.

El fichero a instalar está disponible en los formatos ZIP y TAR-GZ, por comodidad se escogió el de formato ZIP llamado *mysql-connector-java-5.1.5.zip*. Luego de descomprimirlo se deberá copiar solo el fichero *mysql-connector-java-5.1.0-bin.jar* dentro del directorio *jboss_home/server/default/lib/*. Este directorio es en donde se guardan las librerías que utilizará cualquier configuración del JBoss.

3.1.4 Instalación de EJBCA

Finalmente se instalará EJBCA, antes habrá que descargarlo siguiendo el enlace correspondiente de la página *http://www.ejbca.com/download.htm*. En la página de descargas se encontrarán todas las versiones disponibles de EJBCA, es recomendable descargar la última versión. En este caso se descargó la 3.5.2, que tiene disponibles los paquetes *ejbca_3_5_2.zip* y *extra_3_5_2.zip*, el primero contiene la herramienta EJBCA y el segundo contiene el API de una RA externa que se puede implementar dentro de EJBCA, aquí solo se tratará la instalación de EJBCA. Para esto se tendrá que descargar el fichero *ejbca_3_5_2.zip* y, como se hizo con JBoss, descomprimirlo en */usr/local/*. Esto se hace con el comando:

```
unzip ejbca_3_5_2.zip –d /usr/local/
```

De ahora en adelante nos referiremos al directorio */usr/local/ejbca_3_5_2/* como *ejbca_home*.

La instalación de EJBCA creará, por defecto, una autoridad certificadora, un usuario superadministrador y un certificado de servidor SSL. Estos tres elementos son necesarios, uno para empezar a emitir certificados y los otros dos para poder acceder a la admin web. En los ficheros de configuración de la herramienta se puede cambiar la configuración de estos elementos, como por ejemplo el contenido de sus certificados electrónicos.

3.1.4.1 Modificación de los ficheros de configuración

Después de haber descomprimido la herramienta se procederá a modificar los ficheros de configuración del directorio *ejbca_home/conf/*, este directorio es importante porque contiene la mayoría de los ficheros de configuración de las funcionalidades que ofrece EJBCA. Estos ficheros podrán ser personalizados según las necesidades de cada instalación, en un principio cada uno tendrá la extensión **.properties.sample* pero luego de modificarlos ésta se deberá cambiar a *.properties. Para mantener un orden es recomendable primero copiar el fichero a otro con el nombre cambiado para recién luego modificarlo.

Las modificaciones consisten principalmente en cambiar los valores por defecto de las variables definidas dentro de estos ficheros, estas variables contienen datos de la configuración de EJBCA. El cambiar el nombre de los ficheros que se modifiquen es importante porque la herramienta conoce los valores por defecto de las variables y la forma en que se le avisa que se han modificado estos valores es cambiando de nombre a los ficheros, si esto no se hace la herramienta ignorará las modificaciones y tomará los valores por defecto.

Los ficheros a modificar son *database.propiertes.sample*, *ejbca.properties.sample* y *web.properties.sample* porque definen algunos parámetros de instalación importantes, como el gestor de base de datos que se utilizará, las propiedades de la CA que se creará por defecto o las propiedades del servidor HTTP. El primer paso será hacer una copia de cada fichero, los nuevos ficheros tendrán por nombres *database.propiertes*, *ejbca.properties* y *web.properties* respectivamente.

En el fichero *database.properties* se descomentarán las líneas que definen los valores de las variables *datasource* y *database*, como el nombre y prefijo del JNDI (*Java naming and directory Interface*) de la fuente de datos, el gestor de base de datos a utilizar y los datos de acceso a la base de datos que utilizará EJBCA. A continuación se muestran las variables que se modificaron en este fichero con los valores que se le dieron.

```
El nombre y prefijo del JNDI a usar
datasource.jndi-name=EjbcaDS
datasource.jndi-name-prefix=java:/

El gestor de base de datos a usar
database.name=mysql
datasource.mapping=mySQL

La url de acceso a la base de datos y el controlador JDBC
database.url=jdbc:mysql://ovh1.firmaprofesional.com:3306/ejbca_3_5_2
database.driver=com.mysql.jdbc.Driver

Nombre y contraseña del usuario administrador de la base de datos
database.username=ejbca
database.password=ejbca
```

En el fichero *ejbca.properties* habrá que realizar el mismo procedimiento, se descomentarán las líneas que definen los valores de las variables *appserver* y *ca*, como los datos del servidor de aplicaciones que se utilizará y la configuración del certificado de la CA que se creará por defecto. A continuación se muestran las variables más relevantes de este fichero con los valores que se le dieron, solo se modificaron algunas de las variables correspondientes a la configuración básica de dicha CA.

```
El servidor de aplicaciones a utilizar y su ubicación
appserver.type=jboss
appserver.home=${env.JBOSS_HOME}

El nombre y DN de la CA por defecto
ca.name=AdminCA1
ca.dn=CN=AdminCA1,O=EJBCA Sample,C=SE

Las propiedades del token software de la CA
ca.tokentype=soft
ca.tokenpassword=null
ca.tokenproperties=conf/catoken.properties

Algunas características de su certificado
ca.keyspec=2048
ca.keytype=RSA
ca.signaturealgorithm=SHA1WithRSA
ca.validity=3650
ca.policy=null

Las contraseñas para proteger sus keystores en la base de datos
ca.keystorepass=firma69
ca.ocspkeystorepass=firma69
ca.xkmskeystorepass=firma69
ca.cmskeystorepass=firma69
```

De los valores de estas variables se observa que la primera CA que la herramienta creará por defecto tendrá claves software, que su certificado será valido por diez años y que la ubicación de JBoss se obtiene del contenido de la variable de entorno creada anteriormente.

En el fichero *web.properties* se descomentarán las líneas que definen los valores de las variables *java.trustpassword*, *httpserver* y *superadmin*, que ente otras cosas definen la configuración del servidor web y las propiedades del usuario superadministrador creado por defecto. Este usuario es el que tiene los privilegios para acceder a la admin web. A continuación se muestran las variables más relevantes de este fichero con los valores que se le dieron.

```
La contraseña para el keystore de certificados de confianza de EJBCA
java.trustpassword=java69

La contraseña para el keystore del superadministrador
superadmin.password=admin69
superadmin.batch=true

Configuración del certificado SSL
httpsserver.password=server69
httpsserver.hostname=localhost
httpsserver.dn=CN=ovh1.firmaprofesional.com,O=EJBCA Sample,C=SE

Los puertos que JBoss escuchará
httpserver.pubhttp=8080
httpserver.pubhttps=8442
httpserver.privhttps=8443

Las interfaces que JBoss escuchará
httpsserver.bindaddress.pubhttp=0.0.0.0
httpsserver.bindaddress.pubhttps=0.0.0.0
httpsserver.bindaddress.privhttps=0.0.0.0
```

De los valores de estas variables se observa que el certificado de servidor SSL que se creará por defecto tiene como *Common Name* el nombre DNS del servidor en el que se instaló la herramienta, esto es importante para evitar advertencias del explorador de Internet, y que JBoss escuchará en todas las interfaces las peticiones dirigidas a la herramienta.

En el anexo D se adjuntan estos tres ficheros.

3.1.4.2 Utilización de los comandos ANT

Los pasos que se muestran a continuación son los que hay que seguir para terminar la instalación de EJBCA, en estos pasos se usará la herramienta *ant*. Ant es una especie de *make* pero hecho en JAVA, lo que asegura que sea independiente del sistema operativo, es útil porque nos evita realizar las tareas mecánicas y repetitivas propias de las fases de instalación y de construcción de aplicaciones, por eso utilizando el comando *apt-get* es recomendable primero asegurarse si está instalada.

```
apt-get install ant ant-doc ant-gcj ant-optional ant-optional-gcj
```

Luego de haber instalado *ant*, dentro de *ejbca_home* se ejecutará el siguiente comando desde la consola:

```
ant bootstrap
```

Este comando lo que hace es compilar todos los ficheros y agruparlos en ficheros JAR, luego todos los JAR en ficheros WAR, y por último todos los WAR en un fichero EAR, finalmente desplegará este fichero EAR a JBoss. También creará dentro de *ejbca_home*, los directorios *dist/*, *tmp/*, *hwtoken/* y *ocsp-dist/*; dentro de *jboss_home/Server/default/deploy/* los servicios *ejbca.ear*, *ejbca-ds.xml* y *ejbca-mail-service.xml;* y por último la estructura de la base de datos. En este punto podría ocurrir que el siguiente error:

```
OutOfMemoryError: Java heap space java.lang
```

La solución a este error se tratará más adelante, en el punto *"Errores detectados"*.

Después de esto, habrá que iniciar JBoss desde la consola y observar detenidamente su inicio, se le debería ver desplegando todos los servicios que utiliza sin que haya errores. Si no ha habido errores, y sin parar JBoss, desde *ejbca_home* se deberá ejecutar el siguiente comando:

```
ant install
```

Este comando generará los certificados y pares de claves que se necesitan para empezar a utilizar la CA por defecto, y los almacenará, junto con otros datos más, en la base de datos. El comando *ant install* solo se puede ejecutar una vez, la primera vez que EJBCA se instala, si se intenta ejecutar una segunda vez se obtendrán errores.

Si todo ha ido correctamente se habrá creado el directorio *ejbca_home/*p12, este directorio debe contener el repositorio de claves del usuario superadministrador en formato P12 (*superadmin.p12*), el repositorio de claves del servidor SSL (*tomcat.jks*) y el repositorio de claves de las entidades de confianza de EJBCA (*truststore.jks*). El

tema de los repositorios de claves (también llamados *keystores*) en EJBCA se tratará con más detalle más adelante en el siguiente capítulo.

Si la ejecución de *ant install* ha sido exitosa el último paso será volver a desplegar toda la herramienta a JBoss. Antes de esto es importante parar el servidor de aplicaciones (también se puede hacer con CTRL+C). Para volver a desplegar la herramienta, desde *ejbca_home,* se deberá ejecutar:

```
ant deploy
```

Este comando copiará los archivos de configuración modificados y los repositorios de claves creados a JBoss, y volverá a crear los servicios que se crearon con *ant bootstrap*. Si el despliegue se ha realizado correctamente probablemente EJBCA ya esté listo para ser utilizado, para comprobarlo habrá que volver a iniciar JBoss y verificar que no ocurran errores.

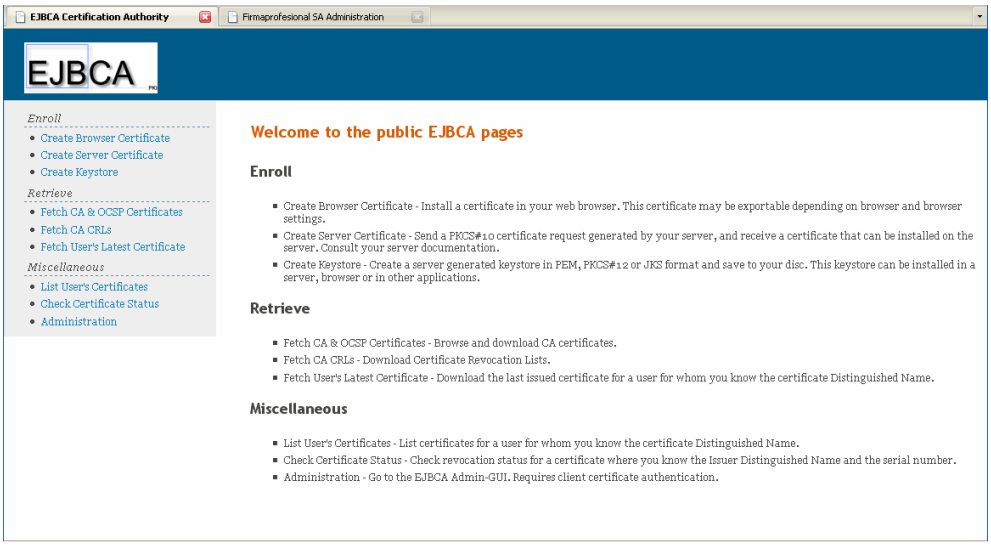
3.1.4.3 Verificación de las interfaces gráficas

Si no ha habido errores en el inicio de JBoss y si todo se ha realizado correctamente se debería poder acceder sin problemas desde un navegador a las interfaces public web y admin web de EJBCA.

A la public web se accede desde *http://ovh1.firmaprofesional.com:8080/ejbca* y en modo seguro desde *https://ovh1.firmaprofesional.com:8442/ejbca*. Esta interfaz no requiere autentificación de usuario y en su página principal tiene tres apartados principales: *Enroll, Retrieve* y *Miscellaneous.* El primero se utiliza para terminar el registro de una entidad final, en este apartado se realiza la creación del par de claves criptográficas y del certificado electrónico de la entidad. Desde el segundo se pueden obtener los certificados y las listas de revocación de las CAs creadas en la herramienta y también los certificados de las entidades finales creadas en la misma. En el tercer apartado es posible saber qué tipos de certificados tiene un usuario y si estos se encuentran aun dentro de su tiempo de validez, este apartado también tiene un enlace para acceder a la admin web.

En esta versión hay un pequeño *bug*, cuando se intenta acceder a la admin web desde la public web que utiliza el puerto 8442, la herramienta intentará utilizar el mismo puerto, esto genera un error de autentificación porque a la admin web solo se accede utilizando el puerto 8443 tal y como está configurado por defecto en *web.properties*.

A continuación se muestra una figura de la apariencia de la public web.



Para acceder a la admin web antes habrá que importar el fichero *ejbca_home/p12/superadmin.p12* en el navegador que se esté utilizando, para esto se necesitará la contraseña que se especificó en *superadmin.* Este fichero es el repositorio de claves del usuario superadministrador que se usará para autenticarse como tal con la admin web, este administrador es el usuario priveilegiado de EJBCA. Si el certificado se ha importado correctamente se podrá acceder desde *https://localhost:8443/ejbca/adminweb/index.jsp*. Se muestra una figura de la apariencia de la admin web.

Esta interfaz tiene cuatro apartados principales que agrupan distintas funciones (*CA, RA, Supervision* y *System functions*), según qué tipo de administrador se haya autenticado algunas de estas funciones aparecerán habilitadas o no. En este caso como la autenticación se ha hecho con el usuario superadministrador, que tiene por nombre SuperAdmin, se tienen todas las funciones habilitadas.

Existe la posibilidad de ejecutar una batería de pruebas de EJBCA, estas pruebas examinan todas las funciones de la herramienta y para esto crea entidades finales, CAs, y certificados de prueba. Por esto es recomendable que si se decide realizarla se exporte antes la estructura y datos de la base de datos a un fichero, así luego de

correr las pruebas se podrá borrar la base de datos, volver a crearla, e importar dicho fichero en ella. Los siguientes comandos son útiles para este propósito:

```
Exportar la base de datos a un fichero
mysqldump --opt -u usuario -p base_de_datos > fichero.sql

Importar la base de datos desde el fichero
mysql -u usuario -p base_de_datos < fichero.sql
```

Si se han realizado personalizaciones en los ficheros **.properties* (no se toman en cuentan las del fichero database.properties) es normal que algunas pruebas de esta batería no se completen correctamente. Esto es porque algunas pruebas están hechas para que terminen exitosamente solo con la configuración por defecto de EJBCA. Con JBoss ejecutándose, desde *ejbca_home* se corren las pruebas con este comando:

```
ant test:run
```

3.2 Inicialización de los logs

Los *logs* son mensajes de aviso de una aplicación por medio de los cuales nos informa sobre cualquier evento que ha sido llevado a cabo en ella. Son importantes porque registran los errores que hayan podido ocurrir en la aplicación y sirven para rastrear el motivo de los mismos. En este apartado se explicará como realizar las configuraciones necesarias para inicializar el sistema de registro de *logs* correctamente.

EJBCA usa el paquete denominado *"Apache Log4j"* para manejar sus *logs* y el servidor de aplicaciones que utilizamos, JBoss, también lo utiliza por defecto para centralizar y administrar tanto sus *logs* como los de las aplicaciones que esté ejecutando. En EJBCA generalmente los *logs* son guardados en la base de datos y a la vez enviados al mecanismo de gestión de *logs* del servidor de aplicaciones.

Log4j considera siete niveles de mensajes de *logging*, de menor a mayor prioridad son: ALL, DEBUG, INFO, WARN, ERROR, FATAL y OFF, según qué nivel se tenga activado se guardarán más o menos mensajes. El primer nivel es el más bajo posible y habilita todos los logs, el último es el nivel más alto y deshabilita todos los logs. El nivel DEBUG se utiliza para guardar mensajes de depuración de la aplicación y se aconseja que esté desactivado en entornos de producción ya que suelen ser muchos, el nivel FATAL se utiliza para guardar mensajes críticos del sistema y generalmente luego de guardarlos la aplicación abortará. Un nivel inferior incluye los mensajes de todos los niveles superiores a él, por ejemplo el nivel INFO incluirá también los niveles WARN, ERROR y FATAL, pero no se cumple lo contrario.

3.2.1 Configuración en EJBCA

Se puede establecer el comportamiento que log4j tendrá con los mensajes de logging propios de EJBCA configurando el parámetro *logging.log4j.config* del fichero *ejbca.properties* con los siguientes valores:

- **False**: No configura log4j. Esto no desactiva log4j solo hace que funcione según lo tiene contemplado el servidor de aplicaciones. Este es el valor por defecto.
- **Basic**: Usa la configuración básica de log4j que es volcar todos los *logs* a la consola.
- **La ruta a un fichero**: El fichero (por ejemplo: *log4j.xml* o *log4j.properties*) tendrá las propiedades de configuración del *logging* para EJBCA, como qué niveles de *logs* almacenar y dónde almacenarlos.

En JBoss log4j se configura en el fichero *jboss_home/server/default/conf/jboss-log4j.xml*, por lo que en *ejbca.properties* se podría escribir:

```
logging.log4j.config= jboss_home/server/default/conf/jboss-log4j.xml
```

Como en JBoss el comportamiento por defecto de log4j se rige por ese mismo fichero, lo anterior es equivalente a:

```
logging.log4j.config=false
```

El darle la ruta del fichero de configuración de log4j al parámetro anterior tiene más sentido cuando se usa un servidor de aplicaciones que no usa log4j por defecto como *Glassfish* o *Weblogic*.

3.2.2 Configuración en JBoss

Para hacer esta configuración habrá que modificar el fichero *jboss-log4j.xml*. Lo que se quiere es guardar todos los *logs* propios de EJBCA en un fichero aparte, por ejemplo, de nombre *ejbca.log*.

Para esto habrá que configurar una entrada *appender* del tipo *RollingFileAppender* y de nombre, por ejemplo, *EJBCA* y que guarde los *logs* en un fichero de salida por ejemplo de nombre *ejbca.log*. Un *appender* no es más que una salida de destino a un fichero, lo que hace es redirigir los *logs* específicos de una aplicación a un fichero determinado. Dentro de él también se pueden configurar otros parámetros como el formato de la fecha y de los mensajes, si los *logs* se añadirán al final del fichero o si reemplazarán los *logs* anteriores cada vez que se ejecute la aplicación y si se creará un fichero distinto cada día o cada hora. El *appender* se muestra a continuación:

```
<appender name="EJBCA"
class="org.jboss.logging.appender.DailyRollingFileAppender">
  <errorHandler class="org.jboss.logging.util.OnlyOnceErrorHandler"/>
  <param name="File" value="${jboss.server.log.dir}/ejbca.log"/>
  <param name="Append" value="true"/>
  <param name="DatePattern" value="'.'yyyy-MM-dd"/>

  <layout class="org.apache.log4j.PatternLayout">
     <param name="ConversionPattern" value="%d %-5p [%c] %m%n"/>
  </layout>

  <filter class="org.jboss.logging.filter.TCLFilter">
     <param name="AcceptOnMatch" value="true"/>
     <param name="DeployURL" value="ejbca"/>
  </filter>
</appender>
```

El orden de los appenders es importante en este fichero, por lo que se debe añadir luego del *appender* de nombre CONSOLE. Este *appender* junto al *appender* FILE son los que controlan los mensajes de *logging* propios de JBoss, CONSOLE utiliza como salida la consola y FILE el fichero *server.log*.

También habrá que definir una entrada *category* que refiera al appender EJBCA, en ella se puede definir el nivel de mensajes de *logging* a guardar así como si se quiere que estos *logs* se dupliquen en los appenders que usa JBoss por defecto. Estas dos opciones se definen cambiando los valores de los parámetros *priority value* y *additivity* respectivamente. A continuación se muestra el category usado:

```
<category name="org.ejbca" additivity="false">
    <priority value="INFO" />
        <appender-ref ref="EJBCA"/>
</category>
```

En este caso, con los valores de *priority value="INFO"* y *additivity="false"* se especifica que se guardarán los *logs* de nivel INFO y que no se duplicarán los logs. También es importante que las entradas *category* aparezcan siempre después de todas las entradas *appender*.

Una alternativa a configurar una entrada *category* específica para EJBCA es configurarle una referencia dentro de la entrada general *root category*. Esta es la entrada *category* principal y en ella está referenciado el *appender* FILE al nivel INFO, es importante que siempre se ubique luego de todas las entradas *category* específicas. Los *appenders* a los que se haga referencia dentro de esta entrada guardarán en su fichero de salida no solo sus *logs* sino también los de FILE. Otra particularidad es que en esta entrada el nivel de *logs* se configura para todos los *appenders* referenciados, por lo que si cambiáramos el nivel de los *logs* de EJBCA también estaríamos cambiando los de FILE, esto no es cómodo por lo que es mejor configurar una entrada *category* específica para cada *appender*. Se muestra como sería la entrada *root category* con la referencia a EJBCA:

```
<!-- Setup the Root category -->
<root>
   <priority value="INFO" />
      <appender-ref ref="EJBCA"/>
       <appender-ref ref="OTRO_APPENDER"/>
</root>
```

Tanto *ejbca.log* como *server.log* se guardan en *jboss_home/server/default/log/*. En este directorio también se guarda un fichero llamado *boot.log*, el cual sólo contiene los *logs* de nivel DEBUG del arranque de JBoss que no aparecen en *server.log*. Esta salida está definida por defecto en el servidor de aplicaciones.

Para que la configuración explicada líneas arriba tenga efecto debe observarse la siguiente línea en la consola luego de iniciar JBoss, esta línea indica qué fichero se usará para configurar log4j.

```
14:15:51,370 INFO [Log4jService$URLWatchTimerTask] Configuring from URL:
resource:jboss-log4j.xml
```

En el anexo D se adjunta el fichero de configuración de logs *jboss-log4j.xml*.

3.3 Errores detectados

Durante el proceso de instalación e inicialización de EJBCA se han encontrado diferentes tipos de errores ocasionados por distintas causas, para resolverlos se tuvieron que tomar distintas medidas. Se ha considerado importante registrarlos para que un futuro, en una eventual reinstalación no se vuelva a tropezar con los mismos errores. Se han tomado en cuenta principalmente los de nivel WARN, ERROR y/o FATAL, además de estos también se muestran algunos de tipo especial. En este apartado se enumerarán los errores encontrados durante la fase de instalación de la herramienta y durante el proceso de inicialización de los mensajes de *logging*.

3.3.1 No espace left on device

Este es un error FATAL que si se produce aborta la acción que se estaba llevando a cabo. Dependiendo de la aplicación que lo genere, el mensaje de error puede aparecer de distintas formas.

- **Error detectado**: Sucedió mientras se utilizaba la consola y luego de ejecutar un comando *ant*. Cuando se realizaban las acciones generadas por el comando, de repente, se abortó la acción y apareció el siguiente mensaje de error:

  ```
  No space left on device
  ```

- **Causa**: No queda espacio suficiente en el disco duro de la unidad que se está usando para realizar la acción deseada.

- **Solución**: Borrar datos de la unidad.

3.3.1.1 Error 28

- **Error detectado**: MySQL respondió con este código de error luego de que se intentara crear una nueva base de datos o una nueva tabla. Es el mismo error que el anterior solo que se produjo mientras se usaba MySQL. Se puede saber que significa este error con *"perror 28"*, se obtendrá lo siguiente:

```
Error code  28:  No space left on device
```

- **Causa**: No queda espacio suficiente en el disco duro de la unidad que se está usando para realizar la acción deseada.

- **Solución**: Borrar datos de la unidad.

3.3.2 OutOfMemoryError

Este es un mensaje de error por falta de memoria obtenido en el fichero *server.log*.

- **Error detectado**: Sucedió a la hora de iniciar JBoss, cuando intenta desplegar los diferentes servicios que tiene asociado, falla y muestra el siguiente error:

```
java.lang.OutOfMemoryError: PermGen space
```

- **Causa**: Los parámetros de memoria configurados para JAVA son por defecto muy bajos, Jboss por defecto configura un máximo uso de memoria de 512Mb lo cual debe ser suficiente para la mayoría de configuraciones. En este caso este máximo no fue suficiente por lo que se tuvo que aumentar este valor límite.

- **Solución**: Modificar la línea que define el máximo y mínimo uso de memoria en JBoss en el fichero *jboss_home/bin/run.conf.* La línea mostrada a continuación es la que define estos parámetros, el número después de *–Xms* define el mínimo de memoria asignada en Megabytes y el de después de *–Xmx* define el máximo de esta memoria. Se aumentó el máximo de memoria a 1024 tal y como se muestra en la línea a continuación.

```
JAVA_OPTS="-Xms128m -Xmx1024m - Dsun.rmi.dgc.client.gcInterval=3600000
-Dsun.rmi.dgc.server.gcInterval=3600000"
```

3.3.3 NotAfter not allowed

Este es un mensaje de error obtenido en el fichero *ejbca.log*.

- **Error detectado**: Sucedió a la hora de crear la autoridad certificadora por defecto del fichero *ejbca.properties*, al tratar de aplicar la configuración de este fichero, falla y muestra el siguiente error:

```
ERROR [org.ejbca.core.model.ca.caadmin.X509CA] notAfter from request
for user 'nobody' is longer than maximum specified in certificate
profile not allowed, using notAfter from certificate profile.
```

- **Causa**: El valor configurado para la validez del certificado de esta CA es menor que lo especificado en el perfil de certificado que se utilizó para crear el certificado de la CA.

- **Solución**: Escoger un periodo de validez de certificado válido.

3.3.4 Failed to stop

Este es un error obtenido en el fichero s*erver.log*.

- **Error detectado**: Sucedió al parar JBoss desde la consola, cuando está parando los servicios que tiene asociados, falla y muestra el siguiente error:

```
ERROR [org.jboss.aop.deployment.AspectDeployer] failed to stop
java.io.FileNotFoundException: /usr/local/jboss-
4.2.2.GA/server/default/tmp/deploy/tmp42141ejb3-interceptors-aop.xml
```

- **Causa**: Aún no se había terminado de iniciar JBoss. Esto es muy probable que suceda cuando por estar modificando los ficheros de configuración de EJBCA se tiene que reiniciar JBoss constantemente.

- **Solución**: Esperar a que JBoss se haya iniciado completamente antes de pararlo.

3.3.5 Errores STDERR

Estos son errores obtenidos en la salida estándar de errores, se almacenan en el fichero *server.log*.

- **Error detectado**: Cuando se inicia el servidor de aplicaciones JBoss aparecen periódicamente estos tres errores:

```
ERROR [STDERR] com.sun.xml.ws.transport.http.servlet.
WSServletContextListener contextInitialized INFO: WSSERVLET12: JAX-WS
context listener initializing

ERROR [STDERR] com.sun.xml.ws.transport.http.servlet.
RuntimeEndpointInfoParser processWsdlLocation INFO: wsdl cannot be found
from DD or annotation. Will generate and publish a new WSDL for SEI
endpoints.

ERROR [STDERR] com.sun.xml.ws.transport.http.servlet.WSServletDelegate
init INFO: WSSERVLET14: JAX-WS servlet initializing
```

- **Causa**: Estos errores tienen como finalidad informar al usuario del estado de de algunos servlets que utiliza JBoss.

- **Solución**: Ninguna. Son mensajes inofensivos que no tienen mayor importancia.

3.3.6 CA token is offline

Este es un error obtenido en el fichero *ejbca.log*, este error puede desencadenar otros tipos de errores.

- **Error detectado**: La CA no puede realizar ninguna acción, no puede emitir certificados, CRLs ni firmar ningún tipo de petición. Se obtuvo el siguiente mensaje:

```
ERROR [org.ejbca.core.ejb.ca.sign.RSASignSessionSession] CA token is
offline for CA E=ca1@firmaprofesional.com,CN=AC Firmaprofesional -
CA1,OU=Jerarquia de Certificacion Firmaprofesional,O=Firmaprofesional
S.A. NIF A-62634068,L=C/ Muntaner 244 Barcelona,C=ES.
```

- **Causa**: El token de la CA estaba desactivado (off-line), y con el token desactivado una CA no puede hacer uso de sus claves de firma.
- **Solución**: Se volvió a activar el token de la CA, en la opción "*View Information"* del enlace *"Basic Functions"* de la admin web,

3.3.6.1 CRL cannot be created

- **Error detectado**: La CA no pudo emitir la CRL correspondiente, al intentar hacerlo se obtuvo este error el cual es consecuencia del error principal anterior.

```
ERROR [org.ejbca.core.ejb.ca.crl.CreateCRLSessionSession] CA Admin_CA,
- 1333369627 is off-line. CRL can not be created.
```

- **Causa**: La CA estaba desactivada (off-line), y en este estado una CA no puede emitir CRLs.
- **Solución**: Se volvió a activar la CA, en la opción "*View Information"* del enlace *"Basic Functions"* de la admin web,

3.3.7 LDAP error

Este es un error obtenido en el fichero *ejbca.log*, ocurrirá si algún publicador LDAP ha sido configurado incorrectamente.

- **Error detectado**: La CA no pudo publicar un certificado o una CRL al servicio de directorios LDAP que tiene configurado. Se obtuvo el siguiente error:

```
ERROR [org.ejbca.core.model.ca.publisher.LdapPublisher] LDAP ERROR:
Error storing CRL (certificateRevocationList;binary) in LDAP
(top;applicationProcess;certificationAuthority) for DN (CN=AC
Firmaprofesional - CA1,c=es).
LDAPException: Object Class Violation (65) Object Class Violation
LDAPException: Server Message: object class 'certificationAuthority'
requires attribute 'cACertificate'
```

- **Causa**: Los atributos y clases a los que se hace referencia en el error no existen en la configuración del publicador LDAP de EJBCA.
- **Solución**: Configurar correctamente los publicadores, definir los atributos y/o clases que se necesitan en la página de edición de publicadores.

3.3.8 Administrator not authorized to resource

Este es un error obtenido tanto en el fichero *ejbca.log* como en *server.log*.

- **Error detectado**: No se pudo acceder al admin web luego de que se creó un nuevo administrador. En el fichero *ejbca.log* se obtuvo este error.

```
ERROR [org.ejbca.ui.web.admin.configuration.StartServicesServlet]
Error init ServiceSession:
java.io.IOException: Authorization Denied
```

  En el fichero *server.log* se obtuvo este error.

```
ERROR [Log4jLogDevice] CEST, CAId : 1272413888, AUTHORIZATION,
EVENT_ERROR_NOTAUTHORIZEDTORESOURCE, Administrator : CLIENTCERT,
Certificate SNR : 3335b13b372f7e8a, CN=NuevaCA,O=FP,C=ES, User : No
user involved, Certificate : No certificate
involved, Comment : Resource : /administrator
```

- **Causa**: Al nuevo administrador se le añadió a un grupo de administradores que estaba asignado a una CA creada recientemente, también se le configuraron unas reglas de acceso. Esta nueva CA se había creado correctamente pero su certificado no había sido añadido al repositorio de claves de confianza de EJBCA

- **Solución**: Añadir el certificado de la nueva CA a dicho repositorio con el siguiente comando:

```
ant –Dca.name=”Nueva_CA” javatruststore
```

3.3.9 Keystore was tampered with, or password was incorrect

Este es un error obtenido en el fichero *server.log*.

- **Error detectado**: Cuando se intentó iniciar el servidor de aplicaciones para que construya EJBCA se obtuvieron los mensajes de error mostrados a continuación, primero un WARN y luego un ERROR. Estos errores hacen referencia a que se tiene una contraseña que no es la que se espera.

```
WARN  [org.jboss.web.tomcat.service.JBossWeb] Failed to startConnectors
LifecycleException:  service.getName(): "jboss.web";  Falló el arranque
del manejador de protocolo: java.io.IOException: Keystore was tampered
with, or password was incorrect

ERROR [org.apache.coyote.http11.Http11Protocol] Error arrancando punto
final (endpoint) java.io.IOException: Keystore was tampered with, or
password was incorrect
```

- **Causa**: Se había cambiado la contraseña del repositorio de claves *truststore.jks* en el fichero */jboss_home/server/default/deploy/jboss-web.deployer/server.xml*. En este fichero se guardan algunas configuraciones del servidor de aplicaciones y las contraseñas que en él se indican deben ser iguales a las declaradas en *web.properties*. Se muestra la parte de *server.xml* en la que se hizo el cambio:

```
<Connector port="8443" address="0.0.0.0"
  axThreads="150" strategy="ms" maxHttpHeaderSize="8192"
  emptySessionPath="true" protocol="HTTP/1.1" SSLEnabled="true"
  scheme="https" secure="true" clientAuth="true"
  keystoreFile="${jboss.server.home.dir}/conf/keystore/keystore.jks"
  keystorePass="server69" sslProtocol = "TLS"
  truststoreFile="${jboss.server.home.dir}/conf/keystore/truststore.jks"
  truststorePass="java69" truststoreType="JKS"
  URIEncoding="ISO-8859-1" />
```

- **Solución**: Cambiar manualmente la contraseña del fichero *server.xml* por la que se le dio a la variable *java.trustpassword* en *web.properties*.

3.3.10 No appenders could be found for logger

Este es un mensaje de alerta obtenido de la configuración de los mensajes de logging de EJBCA.

- **Error detectado**: Se encontró a la salida de un *ant bootstrap* o un *ant deploy*, son dos mensajes de alerta (WARNs). Por lo general un WARN no afecta al correcto funcionamiento de la aplicación, sólo se usa para dejar constancia de ciertos eventos.

```
[jasper2] log4j:WARN No appenders could be found for logger
(org.apache.jasper.compiler.JspRuntimeContext).
[jasper2] log4j:WARN Please initialize the log4j system properly.
```

- **Causa**: En concreto estos dos mensajes aparecen cuando el servidor web de JBoss (jboss-webdeployer.xml) reconoce que log4j ha sido cargado pero advierte que no existe ningún fichero log4j.properties de donde leer su configuración,

- **Solución**: Ninguna. Son mensajes inofensivos que no tienen mayor importancia.

3.3.11 Continuable parsing error

Este es un error de parseo del fichero de configuración de los mensajes de logging de EJBCA, generalmente se presenta en forma de WARNs y no interrumpe el parseo del fichero.

3.3.11.1 Type Id must be unique

- **Error detectado**: Se encontró en la salida del *appender* CONSOLE que escribe en consola.

```
log4j:WARN Continuable parsing error 162 and column 85
log4j:WARN Attribute value "TSA" of type ID must be unique within
the document.
```

- **Causa**: Los nombres de las entradas *appender* (que son los ID de los mismos) dentro del fichero de configuración de log4j deben ser únicos, en caso no lo

sean log4j envía un WARN que indica la línea exacta en la que sucede este problema. En este caso se encontraron dos appenders repetidos de nombre “TSA”.

- **Solución**: Borrar el *appender* repetido para así dejar en el fichero de configuración al *appender* con un ID único, parar JBoss y volver a arrancarlo.

3.3.11.2 Continuable parsing error: The content of element type X must match

- **Error detectado:** Se encontró en la salida del *appender* CONSOLE que escribe en consola.

```
log4j:WARN Continuable parsing error 240 and column 12
log4j:WARN The content of element type "appender" must match
"(errorHandler?,param*,layout?,filter*,appender-ref*)".
```

- **Causa:** Una entrada *appender* debe tener definidas por lo menos las etiquetas errorHandler, param, layout, filter y appender-ref, la última es una referencia al *appender* definida en otra entrada, por ejemplo en una entrada category. Si faltara una de estas se obtendría un WARN indicando la línea del error y otro indicando que al *appender* le falta alguna de estas etiquetas.

- **Solución:** Verificar que al appender no le falte ninguna de estas etiquetas, y si le faltara alguna escribirla. A veces el problema puede ser que sin querer se haya comentado alguna de las etiquetas.

3.3.12 Fatal parsing error: Could not parse url

Este es un error de parseo del fichero de configuración de los mensajes de logging de EJBCA que sí interrumpe el parseo del fichero.

- **Error detectado**: Se encontró en la salida del appender CONSOLE que escribe en consola.

```
log4j:WARN Fatal parsing error 469 and column 17
log4j:WARN The string "--" is not permitted within comments.
log4j:ERROR Could not parse url [resource:jboss-log4j.xml]
```

- **Causa**: Dejar caracteres en el fichero de configuración que log4j no entiende. Esto genera un WARN indicando la línea del error y otro indicando el carácter que no se entiende, el aviso termina con un ERROR indicando que se ha interrumpido el parseo del fichero. En este caso el error se produjo por haber dejado los caracteres “--” dentro de un comentario ya que se pueden confundir con los caracteres “-->” que simbolizan el final del comentario.

- **Solución**: Verificar que no se hayan dejado caracteres extraños en el fichero de configuración.

CAPÍTULO 4: CONFIGURACIÓN

Este capítulo detalla el procedimiento de configuración de la herramienta que se ha seguido para, primero, crear la *Jerarquía de Certificación de Firmaprofesional*, lo cual incluye la creación de las autoridades certificadoras y de todos los tipos de certificados reconocidos de entidad final que emite la empresa, y, segundo, llevar a cabo la simulación del proceso de migración de la jerarquía creada, esto se realizará importando toda la configuración anterior a una nueva instalación de EJBCA. Antes de abordar estos dos importantes apartados se explica, en un primer apartado, la configuración y uso de funciones básicas de la herramienta que permite su personalización y entendimiento.

Para realizar las configuraciones se usará la admin web y la CLI, a la primera se accederá con el usuario Superadmin que se creó durante la instalación y que tiene privilegios de superadministrador. Este usuario puede realizar cualquier función y es el que se utilizará para realizar las primeras configuraciones. A la segunda se accederá de manera local o de manera remota accediendo vía SSH al ordenador en el que está instalada la herramienta.

4.1 Configuraciones previas

En este apartado se detallará la configuración de las características generales y de algunas funcionalidades relevantes de la herramienta, también se explicarán algunos procedimientos básicos para la correcta administración de la herramienta.

4.1.1 Configuración de la admin web

El primer paso es configurar las características generales del sistema, en el apartado *"System Functions"* de la admin web se encuentra *"System Configuration"*. Este enlace lleva a la página de configuración web de EJBCA, en ella se pueden modificar tanto características de apariencia como las de funcionamiento de la herramienta. En la siguiente tabla se describe el significado de cada opción de configuración y cómo se ha configurado.

CONFIGURACIÓN WEB	
Opción	**Descripción**
Title	Se escribe el nuevo título del sitio web: "Firmaprofesional SA Administration".
Head Banner	Se pueden cambiar los banner por defecto a algún fichero JSP o HTML que se especifique. Esto se tratará más adelante.
Foot Banner	
Enable End Entity Profile Limitations	Si se quiere restringir qué entidades finales podrá controlar una RA. Habilitado.
Enable Key Recovery	Si se quiere poder recuperar las claves privadas en caso de pédida. Deshabilitado.
Issue Hardware Tokens	Si se quiere poder emitir tokens hardware. Habilitado.
Hard Token Data Encrypt CA	Se especifica con que CA se encriptarán los datos sensibles de los tokens hardware. De momento dejar en "No encryption".
Use Approval Notifications	Si se quiere enviar emails cada vez que se procese una acción de aprobación. No se usarán acciones de aprobación.
Email Address to Approving Administrators	Se escribe un alias para todos los administradores q procesen aprobaciones. Deshabilitado.
Approval Notification From Address	Se escribe el email desde el que se envian las notificaciones. Deshabilitado.

También se tiene una página de configuración de las preferencias de visualización de la admin web a la que se accede mediante el enlace *"My Preferences"* del apartado *"System Functions"*, cada administrador puede establecer sus propias preferencias. En la siguiente tabla se describe brevemente cada preferencia y cómo se ha configurado.

PREFERENCIAS DE ADMINISTRADOR	
Opción	**Descripción**
Preferred Language	Se selecciona el lenguaje a usar por defecto en toda la admin web. Se seleccionó EN (Inglés).
Secondary Language	Se selecciona el lenguaje secundario a usar en caso no se encuentre traducción para un texto en el lenguaje por defecto. Se seleccionó ES (Español).
Theme	Se puede escoger el tema (fichero css que define fuentes y colores) de la página. Estos se guardan en ejbca_home/src/adminweb/themes/. No se modificó.
Number of Records per Page	El número de records que se mostrarán por página. Se dejó en 25.

4.1.1.1 Personalización de los banners

Los *banners* pueden ser ficheros HTML o JSP, la admin web usa uno de encabezado (*head_banner.jsp*) y otro de pie de página (*foot banner.jsp*), ambos se pueden personalizar ya sea creando nuevos o modificando los actuales. Los nuevos *banners* deben guardarse en *ejbca_home/src/adminweb/banners*/ y las imágenes que utilicen en *ejbca_home/src/adminweb/images/*, en estos directorios también se encuentran los *banners* actuales y las imágenes que utilizan. Luego de modificar algún *banner* hay que hacer un *ant deploy* para poder apreciar los cambios.

La forma más sencilla es modificar los *banners* actuales, porque así no se tiene que modificar el código de los ficheros solo cambiar los parámetros que utiliza. A continuación se explica qué hace cada *banner* y cómo modificarlos:

- **Head banner**: *head_banner.jsp* muestra la imagen por defecto de encabezado de página de EJBCA (de nombre *ejbcaheader.jpg*). Para cambiarla se debe guardar la nueva imagen, en el directorio correspondiente, con el mismo nombre.

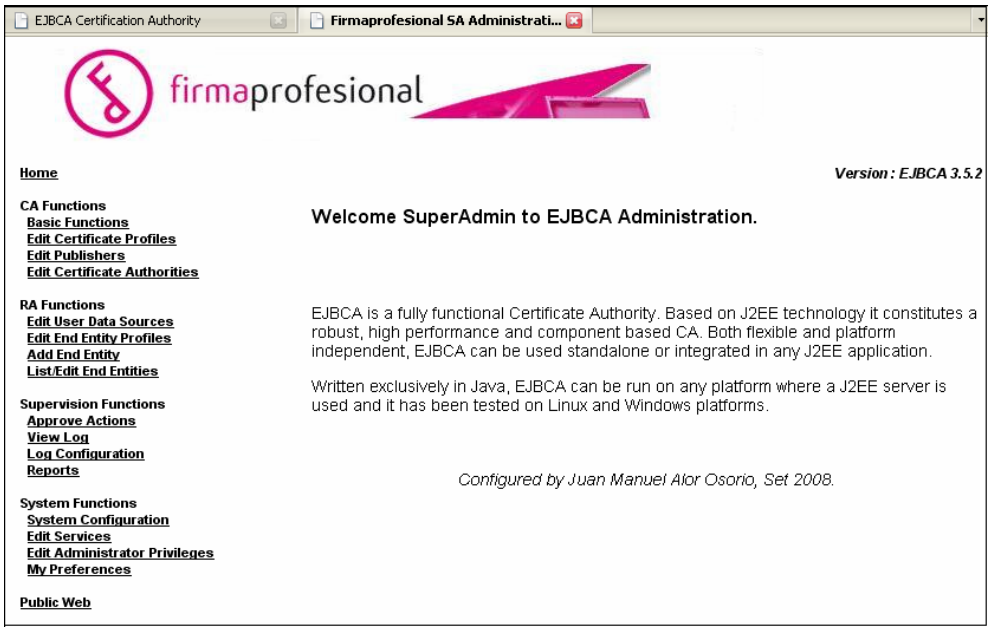
- **Foot Banner**: *foot_banner.jsp* muestra en el pie de página de la admin web la cadena de texto definida por el parámetro MADEBYPRIMEKEY en los ficheros de idiomas *ejbca_home/src/adminweb/languages/languagefile.XX.properties*. Para cambiar el texto por defecto se debe añadir un nuevo parámetro en estos ficheros (basta modificar los ficheros en el que *XX* vale *en* y *es*) con el texto deseado y llamarlo desde *foot_banner.jsp*. De esta manera:

```
En languagefile.en.properties:
# Se añade el parámetro MADEBYJM
MADEBYJM = Customized by Juan Manuel Alor Osorio, Set 2008.

En foot_banner.jsp:
<% //Se modifica la siguiente línea%>
<div align="center" id="footer">
      <i><%=ejbcawebbean.getText("MADEBYJM") %></i></div>
```

Luego de haber personalizado los *banners* se tendrá una nueva apariencia en la admin web. En realidad también es relativamente sencillo personalizar el look and feel de la public web, en *ejbca_home/src/publicweb/publicweb* se encuentran todos las páginas públicas y están escritas en JSP, se pueden modificar directamente pero para

apreciar los cambios luego de la modificación hay que hacer un *ant deploy*. La nueva apariencia de la admin web se puede apreciar en la figura siguiente.



Es recomendable que cualquier personalización de ficheros que se haga se guarde en un árbol de directorios aparte de ejbca_home para no perderlos a la hora de realizar actualizaciones, el directorio ejbca-custom es el que EJBCA ha pensado para guardar estos cambios aunque de él se hablará más adelante.

Los ficheros *ejbcaheader.jpg*, *foot_banner.jsp, languagefile.en.properties* y *languagefile.es.properties* se incluyen en el anexo D.

4.1.2 Configuración de Publishers

Cómo se ha visto anteriormente, en EJBCA existe algo que llaman Publishers. Con este nombre se hace referencia tanto al encargado de realizar la publicación de los certificados y CRLs emitidos como a la ubicación central donde estos se almacenan. EJBCA ya tiene implementados lo dos tipos de publishers más utilizados: LDAP y Active Directory de Windows, se pueden crear publishers de cualquiera de estos dos tipos de forma no muy complicada. Si se necesita un tipo de publisher más personalizado se puede desarrollar, hay información de cómo hacerlo en el directorio *ejbca_home/src/java/org/ejbca/core/model/ca/publishers/* incluso hay uno que puede servir de ejemplo llamado *DummyCustomPublisher*.

La creación de uno nuevo se hace desde el enlace *"Edit Publishers"* del aparatado *"CA Functions"* de la admin web, en esta página se pueden crear, renombrar, borrar y editar publishers. Para propósitos de este proyecto se creó un publisher de tipo LDAP llamado *LDAP_FP.* Dentro de la página de configuración del publisher se deben establecer los parámetros de la configuración LDAP del servidor en el que se ha instalado EJBCA, esto quiere decir que previamente se le ha debido instalar y configurar correctamente el servicio de directorios LDAP lo que implica haber creado una estructura de directorios y, si es el caso, un schema propio.

En la siguiente tabla se muestra cómo se configuró el publisher, a algunas opciones se le ha dejado el valor por defecto y a otras se les ha modificado, se detallan solo las que se han modificado.

CONFIGURACIÓN DE PUBLISHERS	
Opción	**Descripción**
Publisher Type	El tipo de publisher a configurar, para el caso de LDAP elegir *LDAP v3 publisher.*
Hostname:	El hostname del servidor en el que está instalado LDAP.
Port	El puerto por el cual el servidor LDAP escuchará peticiones, si se usa SSL será el 636, si no el 389.
Use SSL	Habilitar si usa SSL.
Base DN	Se ingresa la locación base del directorio LDAP donde se publicarán los objetos (certificados o CRLs).
Login DN	El DN del usuario admin con permiso de escritura para publicar los certificados.
Login password	El password de este admin DN.
Confirm password	
Create Nonexisting Users	Define si un objeto LDAP debe ser creado por EJBCA si no existe cuando se publica el certificado. Habilitado.
Modify Existing Users	Define si un atributo de un objeto LDAP existente (como el email) será modificado si una entrada es actualizada por ejemplo con nuevo certificado. Habilitado.
Add multiple certificates per user	Si un usuario puede o no tener más de un certificado. Deshabilitado. En este caso si se publica un nuevo certificado el anterior se reemplazará con el nuevo.
Remove certificates when revoked	Si un certificado es revocado se borrará la entrada del mismo. Habilitado.
Remove ldap user when certificate revoked	Si un certificado es revocado se borrará la entrada del usuario al que pertenece. Deshabilitado.
User Object Class	Estos campos permiten personalizar los atributos usados según el LDAP schema que se haya definido en la configuración del servidor LDAP. Se dejan los campos por defecto.
CA Object Class	
User Certificate Attribute	
CA Certificate Attribute	
CRL Attribute	
ARL Attribute	
LDAP location fields from cert DN	Se escogen los campos del DN del certificado que añadidos a los del Base DN formarán el DN que se usará para la publicación en LDAP.

Cabe recalcar que el modo de operación de un publisher LDAP consiste en buscar el DN del objeto a publicar en LDAP. Cuando EJBCA publica un certificado lo hace creando un objeto, que es realmente lo que se guarda en LDAP. Para esto primero se construye el DN añadiendo el *"Base DN"* a los campos seleccionados en *"LDAP location fields from cert DN"*, y con él comprueba si la entrada existe en LDAP para saber si debe crear una nueva o modificar la existente. Si el DN de publicación en LDAP de un usuario no contiene todos los campos de su DN de usuario en EJBCA, los campos que falten serán guardados como atributos suyos.

Para empezar a publicar certificados usando el publisher creado se debe crear un perfil de certificado que lo tenga seleccionado, si previamente se ha configurado correctamente el publisher éste podrá ser seleccionado.

4.1.3 Diseño de Certificados

Antes de empezar es importante diferenciar entre certificados de CAs (o SubCAs) y certificados de entidad finales. Para crear los certificados de las primeras solo basta con configurar un Perfil de Certificado (y realizar alguna configuración más, esto se explica mejor más adelante) pero para crear los certificados de las entidades finales se necesita además definir un Perfil de Entidad Final. En resumen, el perfil de certificado configura qué campos y extensiones tendrá el certificado, y el perfil de entidad final pule las características del certificado más específicas a la entidad final (como los campos que tendrá el *Subject DN* o el *Subject Alternative Name*). Al proceso de configurar un Perfil de Certificado (Certificate Profile) y un Perfil de Entidad Final (End

Entity Profile) se le llama diseñar un certificado de entidad final, esto solo se puede hacer desde la admin web.

En caso que no se necesite crear un certificado de usuario con características especiales existen algunos perfiles de certificados ya creados (los que tienen entre paréntesis FIXED) que se pueden usar para diseñar los primeros certificados.

4.1.3.1 Configuración de un Perfil de Certificado

En "*Edit Certificate Profiles"* del apartado "*CA Functions"* de la admin web se puede editar, borrar y crear nuevos perfiles de certificado. Existe la posibilidad de crear un nuevo perfil utilizando alguno de los existentes como plantilla, para esto se tiene que escribir el nuevo nombre de perfil y pulsar el botón *"Use Selected as Template"*, esta opción es útil si se quieren crear nuevos perfiles muy parecidos a uno ya creado. Para cambiar el nombre de un perfil se le tiene que seleccionar, escribir el nuevo nombre y pulsar el botón *"Rename Selected"*.

Cuando se haya entrado a la página de edición del perfil seleccionado se tendrá una lista de opciones de configuración por seleccionar y/o rellenar. Estas opciones permitirán configurar los atributos o características especiales de los campos básicos y/o extensiones (como las extensiones x.509v3) que tendrá el certificado de entidad final. Las extensiones *Key Usage* y el *Extended Key Usage* son especialmente importantes porque definen el uso y uso extendido de la clave pública que contiene el certificado. En el anexo C se listan los usos y usos extendidos que soporta EJBCA para la clave de un certificado.

Los campos básicos y extensiones de un certificado son explicados a profundidad en la recomendación *RFC 3280* que define los perfiles de certificado, también se puede encontrar una breve explicación de ellos en la wiki de la herramienta. EJBCA también sigue esta recomendación, esto se refleja en que en las opciones de configuración de perfil solo acepta valores que cumplan lo especificado en la recomendación y en que los valores por defecto de estas opciones también la siguen.

En la tabla de nombre *OPCIONES DE CONFIGURACIÓN DEL PERFIL DE CERTFICADO* del anexo E se explica el significado de las opciones de configuración de la página de edición del perfil de certificado.

En el anexo B se incluye una imagen de la apariencia de la página de edición del perfil de certificado.

4.1.3.2 Configuración de un Perfil de Entidad Final

En *"Edit End Entity Profiles"* del apartado *"RA Functions"* de la admin web se puede editar, borrar y crear nuevos perfiles de entidad final. Tal y como en el caso de los perfiles de certificado también es posible renombrar perfiles y crear un nuevo perfil usando uno antiguo como plantilla, se siguen los mismos pasos.

Cuando se haya entrado a la página de edición del perfil seleccionado se tendrá una lista de opciones de configuración por definir. Estas opciones permitirán configurar algunos campos básicos y/o extensiones del certificado X.509 que definirán exactamente a la entidad final propietaria del certificado a crear con este perfil. Dos de estos campos especialmente importantes son los que especifican el *DN* (*Distiguished Name*) y el *Subject Alternative Name* de la entidad final. El primero es un campo

básico y especifica un nombre único que identifica al dueño del certificado y que puede estar compuesto por varios campos más, el segundo es una extensión que especifica los nombres alternativos que puede tener el sujeto y puede estar formado por más de un campo. En el anexo C se lista y explica el significado de los campos de DN y nombre alternativo que soporta EJBCA. Se puede encontrar una breve explicación de algunos de los campos presentes en un perfil de entidad final en la wiki de EJBCA.

En la tabla de nombre *OPCIONES DE CONFIGURACIÓN DEL PERFIL DE ENTIDAD FINAL* del anexo E se explica el significado de las opciones de configuración de la página de edición del perfil de entidad.

En la página de edición del perfil de entidad final algunas opciones de configuración tienen una casillas (de nombres *Use*, *Default*, *Required, Modifiable*) para habilitar o deshabilitar. Cada casilla cumple una función específica y modifica el comportamiento de las opciones en la página de adición de entidad final.

- **Use**: Si se habilita esta casilla indica que esta opción aparecerá como un campo más en la página de adición de entidad final cuando la RA utilice este perfil. Si no se habilita la casilla la opción no aparecerá.

- **Default**: Sólo si la casilla anterior está habilitada se puede habilitar esta casilla que lo que hace es que la opción aparezca habilitada por defecto en la página de adición de entidad final. Si no se habilita la casilla la opción aparecerá deshabilitada por defecto.

- **Required**: Esta casilla hace que la opción aparezca habilitada y no se pueda deshabilitar en la página de adición de entidad final. Si en el perfil de entidad final la opción tiene una caja de texto (como en el caso de *Username*), en la página de adición de la entidad final la caja de texto tendrá que estar rellenada para que se pueda añadir la entidad final con éxito.

- **Modifiable**: Si la opción del perfil de entidad final tiene una caja de texto asociada el valor que se escriba en ella aparecerá como el valor por defecto en la página de adición de entidad final, si la casilla *Modifiable* no está habilitada la RA no podrá modificar este contenido. Si se escribe más de un valor utilizando el carácter ";" como separador en la página de adición de entidad final estos valores aparecerán dentro de una caja de selección de texto.

En el anexo B se incluye una imagen de la apariencia de la página de edición del perfil de entidad final.

4.1.4 Creación de grupos de administradores

EJBCA contempla diferentes tipos de administradores, dentro de la lógica de la herramienta cada tipo está pensado para que lleve a cabo tareas distintas según el componente de la herramienta (CA o RA) del que se encargue, con el objetivo de distribuir las tareas de administración y aumentar la seguridad. Los tipos de administradores en orden de mayor a menor importancia son: Superadministrador, Administrador de CA, Administrador de RA y Supervisor. El administrador de mayor importancia puede realizar en general todas las tareas del de menor importancia. A grandes rasgos se especifican las funciones de cada tipo en la siguiente tabla.

FUNCIONES DE LOS ADMINISTRADORES	
Tipo	**Funciones**
Superadministrador	Este adminstrador tiene acceso a todo el sistema y puede: - Editar la configuración de todo el sistema - Crear publishers - Crear, editar, eliminar, activar y desactivar CAs - Crear superadministradores y administradores de CA Puede realizar las tareas de
Adminitrador de CA	Puede administrar una CA y puede: - Crear administradores de RA - Administrar perfiles de certificado y de entidad final de usuarios finales - Configurar las opciones de almacenamiento de logs de la CA Realizar tareas de administrador RA y de supervisor
Adminitrador de RA	Puede administrar una RA y puede: - Añadir, consultar, editar y eliminar usuarios finales - Consultar, editar, eliminar y revocar certificados de usuarios finales Puede realizar tareas de supervisor
Supervisor	Puede supervisar CAs y RAs, pueden: - Ver los logs para observar quién ha hecho qué. - Consultar datos de usuarios y de sus certificados

Normalmente dentro de la empresa solo debe existir un superadministrador, cada administrador de CA solo puede administrar la CA que se le ha sido asignada y por lo general una CA sólo tiene un administrador de CA. Una CA puede tener más de un solo administrador de RA. Por último los supervisores normalmente se asignan también por CA y pueden observar todas las actividades de los administradores de esa CA.

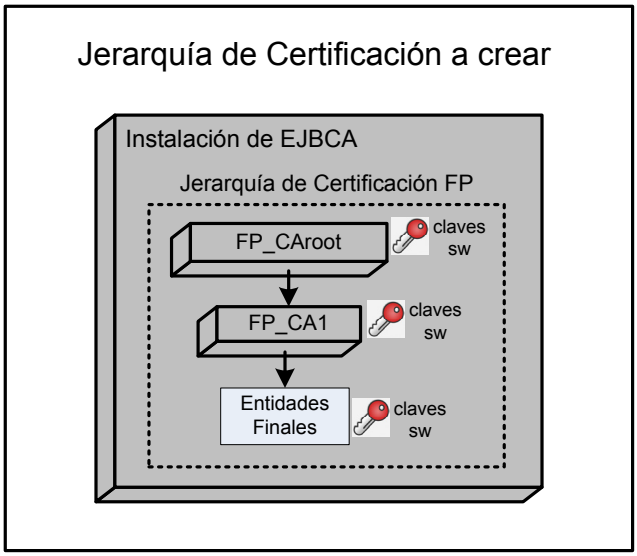
Antes de crear a un administrador (sea del tipo que sea) se debe primero crear un *"Administrators Group"* desde la admin web *(System Functions / Edit Administrators Priviledges)* y asignarlo a una de las CA de EJBCA. Luego hay que añadir los criterios de selección de administradores, por ejemplo: "Serán administradores todos las entidades finales que tengan CN=admin". Por último en el link *"Edit Acces Rules"* hay que escoger que el rol (tipo de administrador) que tendrán las entidades finales pertenecientes a este grupo. Para finalizar se deben crear las entidades finales que cumplan con los criterios de selección del grupo, esto comprende las tareas de creación del perfil de certificado y del perfil de entidad final de tal tipo de administrador y la propia tarea de dar de alta a las entidades finales. Los tutoriales de administración de EJBCA explican claramente y de forma gráfica cómo se realiza la creación de cada tipo de administrador.

A efectos de este proyecto no es imprescindible crear diferentes tipos de administradores ya que para utilizar las funcionalidades de EJBCA que utilizamos basta con el usuario Superadmin que se crea al instalar la herramienta. Aunque no estaría de más crear un administrador de CA que se encargue de administrar la CA subordinada principal y así reservar el acceso como Superadmin para tareas excepcionales. En un futuro, y pensando en el escenario real, si se habilitara el acceso desde la RA de Firmaprofesional a la CA sería interesante crear usuarios administradores de RA para cada operador que tenga acceso a ella.

4.2 Configuración de la Jerarquía de Certificación de FP

Como se explicó en el capítulo 1, la *Jerarquía de Certificación Firmaprofesional* está formada principalmente por una CA raíz llamada *"Autoridad de Certificación Firmaprofesional CIF A62634068",* y por una CA subordinada principal llamada *"AC Firmaprofesional - CA1"*, esta última es autoridad principal que se encarga de emitir certificados de entidad final. En este apartado se explicarán los pasos a seguir para crear y configurar estas CAs en EJBCA, y también los necesarios para que puedan crear y expedir todos los tipos de certificados reconocidos de entidad final que emite la

empresa. Tanto las CAs como las entidades finales, que serán dueñas de los certificados a crear, se crearán usando *soft tokens*, esto quiere decir que las claves de firma y de encriptación serán almacenadas en un fichero PKCS#12 dentro de la base de datos. A continuación se muestra una figura de la jerarquía de certificación que se creará



El objetivo final de la creación de esta jerarquía es que EJBCA pueda emitir certificados de entidad final que cumplan las políticas de certificación de Firmaprofesional. Este proyecto solo se centrará en los certificados reconocidos según la ley 59/2003, ya que al estar regulados por la administración pública son los más delicados de crear. Para alcanzar el objetivo anterior es necesario que los certificados de las CAs creadas sean lo más parecido posible a los verdaderos, críticamente el de la CA subordinada principal que es la que emitirá los certificados de entidad final ya que algunos campos de su certificado aparecerán en los certificados de entidad final que emitan (el DN y el nombre alternativo). Además en el siguiente capítulo solo será necesario importar esta autoridad principal.

4.2.1 Creación de las autoridades certificadoras

En EJBCA para crear una autoridad certificadora se necesita crear antes un perfil de certificado que la defina. Si el contenido del certificado de la CA no es importante se pueden utilizar los perfiles de certificado que vienen creados por defecto (los que tienen la palabra FIXED entre paréntesis), si este es el caso los perfiles ROOTCA y SUBCA servirán para crear la CA raíz y la CA subordinada respectivamente. Sin embargo para cumplir los objetivos de este proyecto se necesitan definir dos perfiles de certificados que emulen correctamente el contenido del certificado de cada CA de Firmaprofesional.

Antes de crear las autoridades certificadoras es necesario primero ahondar en el contenido de los certificados de las CAs de Firmaprofesional y luego crear los perfiles de certificado que me permitan obtener el mismo contenido en los certificados de las CAs que se crearán.

4.2.1.1 Contenido de los certificados de las CAs de FP

A la estructura del contenido de los certificados electrónicos también se le suele llamar perfil de certificado, los perfiles de certificado de las CAs de Firmaprofesional siguen el estándar X.509v3 definido en el *RFC 3280* y son coherentes con lo dispuesto en las normas *ETSI TS 101 862* (*"European profile for Qualified Certificates"*) y *RFC 3739* (*"Qualified Certificates Profile"*).

En la tabla de nombre *CONTENIDO DE LOS CERTFICADOS DE LAS CAs DE FP* del anexo E se muestra el contenido de los campos básicos y de las extensiones del certificado electrónico de cada CA.

El contenido de algunos campos básicos y extensiones de estos certificados será imposible de reproducir de manera exactamente idéntica en los certificados que se creen. Esto sucede con los campos *Serial Number*, *notBefore*, *notAfter*, *Private Key Usage Period*, *Authority Key identifier* y *Subject Key Identifier*. El motivo de esto para el caso de los cuatro primeros es que son campos cuyo valor depende del momento exacto en que se expide el certificado. En el caso de los dos últimos campos el problema es que su valor depende directamente del par de claves criptográficas y éstas, a menos que se importen, no se pueden duplicar.

4.2.1.2 Configuración de los perfiles de certificado

Para crear unas CAs cuyos certificados electrónicos tengan la misma información que la que se ha mostrado en el punto anterior se tienen que crear los perfiles de certificado que definan esta información. En total se crearon dos perfiles de certificado, uno para la CA raíz y otro para la CA Subordinada, el nombre que se les dio a los perfiles fue *FP_CARoot* y *FP_CASubordinada* respectivamente.

En la tabla de nombre *CONFIGURACIÓN DE LOS PERFILES DE CERTIFICADO DE LAS CAs CREADAS* del anexo E se muestran los valores que se le dieron a las opciones de configuración más importantes de los perfiles de certificado creados.

De esta tabla vale la pena comentar brevemente porqué a la opción "*Available CAs"* se le da el valor *AnyCA* y porqué no se eligió solo *FP_CA1*, si se hubiera elegido *FP_CA1* solo ella hubiera podido utilizar el perfil y la idea era que este perfil sirviera para que cualquier CA pudiera crear una autoridad subordinada con las características de cada perfil. Elegir este valor también es especialmente útil a la hora de exportar e importar perfiles de certificado, esto se explicará con más detalle más adelante.

Otro aspecto interesante a comentar es que si se deshabilita el uso de alguna extensión no quiere decir que se esté desactivando la funcionalidad de la herramienta que pueda tener asociada esta extensión. Por ejemplo, aunque se deshabilite el uso del *CRL Distribution Point* la funcionalidad de generación de CRLs de la herramienta seguirá estando activada, solo sucederá que cuando se emita el certificado esta extensión no aparecerá en él.

4.2.1.3 Creación y configuración de las CAs

Para crear una CA hay que seguir el enlace *"Edit Certificate Authorities"* del apartado *"CA Functions"* de la admin web, así se llega a una página intermedia en la que se pueden editar, crear, renombrar y eliminar autoridades certificadoras. Para crear una basta con escribir el nombre que se le quiere dar, es recomendable que no contenga

espacios, dentro de la caja de texto de la sección *"Add"* y presionar el botón "*Create"*, esta acción abre la página de creación de CAs en la que se configurarán los atributos de la CA.

En el anexo B se incluye en una figura la apariencia de la página de creación de CAs, en ella se pueden ver los atributos a configurar y los valores por defecto de esta página.

Se crearon las autoridades certificadoras de nombres *FP_CAroot* y *FP_CA1* correspondientes a la CA raíz y a la CA subordinada. En caso se les quiera cambiar alguna propiedad se les puede editar luego de creadas, en la página de edición de CAs también se puede revocar el certificado de la CA, renovar su par de claves (para esto se necesita el código de autenticación), renovar su certificado y exportar la CA.

En la tabla de nombre *CREACIÓN DE LAS AUTORIDADES CERTIFICADORAS* del anexo E se muestra el significado de las opciones de configuración de los atributos de las CAs y el valor que se le dio a cada una de estas opciones para lograr la configuración que proporcione los certificados más parecidos a los de las autoridades certificadoras de Firmaprofesional.

Algunas de las opciones configuradas son especialmente importantes para conseguir un determinado comportamiento de las CAs. Como se explicó al inicio Firmaprofesional sigue las directrices de los estándares de certificación, por eso las codificaciones elegidas son las UTF8.

Es importante crear la CA con el orden del DN configurado según X.500 ya que Firmaprofesional usa este orden es sus certificados, por eso se habilitó la opción *Use LDAP DN order*. Siguiendo la referencia X.500 los campos del DN se ordenarían así: *"CN=Juanma, O= EJBCA, C=ES"*, y según la referencia LDAP así: *"C=ES, O=EJBCA, CN=Juanma"*. Si se ha creado la CA con un orden de DN no se deberá cambiar este orden luego, si se hiciera podrían ocurrir fenómenos extraños. En la práctica, si bien es cierto, pocas aplicaciones son sensitivas al orden del DN es mejor configurar correctamente esta opción para así evitar cualquier posibilidad de error.

Como sucedía en el caso de los perfiles de certificado, si a la hora de configurar una CA se deja en blanco la opción *"Default CRL Distribution Point"* lo único que sucederá será que esta extensión no aparecerá ni en su certificado ni en la CRL que emita. La diferencia es que el valor que se configure aquí reemplazará al posible valor configurado en el perfil de certificado y que éste valor también aparecerá en la extensión *"Issuing Dsitribution Point"* de la CRL que emita esta CA.

4.2.1.4 Limitaciones de los certificados de CA

Para poder examinar si los certificados de CA creados tienen alguna limitación primero hay que saber como descargarlos. Un administrador puede obtener los certificados de CAs accediendo a la página de funciones básicas de las CAs, esto se consigue siguiendo el enlace *"Basic Functions"* del apartado *"System Functions"* de la admin web. En esta página se listan las autoridades certificadoras que maneja EJBCA y se pueden obtener los certificados digitales (en formato PEM, JKS y CRT) y las listas de revocación más actuales de las mismas. La página también muestra el estado de la CA y la fecha de expiración de la CRL actual, en caso se necesite se puede hacer que emita una nueva CRL manualmente. Aquí también se puede activar o desactivar una CA entrando a su enlace *"View information"*, esto es útil porque cada vez que se edita una CA automáticamente se desactiva por lo que es necesario activarla manualmente.

Por otro lado los usuarios comunes de la herramienta pueden obtener los certificados de las CAs siguiendo el enlace *"Fetch CA & OCSP Certificates"* de la public web, este enlace lleva a una página en la que se listan las CAs que tiene la herramienta y sus respectivos certificados.

Como ya se explicó anteriormente algunos campos de los certificados de las CAs de Firmaprofesional eran imposibles de reproducir de manera idéntica (*Serial Number*, *notBefore*, *notAfter*, *Private Key Usage Period*, *Authority Key identifier* y *Subject Key Identifier*). Además de estos hubo extensiones X.509v3 que no se pudieron configurar para que aparezcan en los certificados creados, como la *X509v3 Private Key Usage Period* y los componentes *DirName* y *Serial* de la extensión *X509v3 Authority Key Identifier*. Esto supone una diferencia con los certificados de Firmaprofesional producida debido a que la herramienta no soporta estas extensiones. Sin embargo, esta diferencia no es crítica porque los certificados de CA (creados con EJBCA) sólo servirán para emitir los certificados de entidad final que cumplan con las políticas de certificación de Firmaprofesional y al fin y al cabo son estos certificados los importantes para el objetivo final de este proyecto.

La extensión *X509v3 Issuer Alternative Name* tampoco es soportada por EJBCA pero se añadirá como una extensión básica (*Custom Basic Extensions*), el cómo hacerlo se explicará más adelante.

Es importante resaltar que los campos del certificado de la CA subordinada creada que interesaba que sean idénticos a los verdaderos, el campo básico *Subject DN* y la extensión *Subject Alternative Name*, se han logrado crear de manera idéntica al certificado original.

4.2.2 Los certificados de entidad final de FP

Como se ha dicho antes, la creación de certificados de entidad final que cumplan las políticas de certificación de Firmaprofesional es el objetivo final de este apartado y uno de los objetivos finales de este proyecto. Es importante porque la consecución de este objetivo garantiza que con la herramienta EJBCA será posible emitir certificados reconocidos idénticos a los que la empresa ha venido emitiendo hasta ahora, con lo que su actividad no se verá afectada y la futura migración será exitosa.

Si bien es cierto este proyecto se centra solo en las políticas de los certificados reconocidos por ley, el procedimiento para crear los demás tipos de certificado que emite Firmaprofesional será muy parecido al de los certificados reconocidos, por lo que el crearlos más adelante no será problema. El prestarle atención solo a los certificados reconocidos por la ley 59/2003 tiene sentido porque el uso y la correcta expedición de estos certificados son procedimientos críticos porque son regulados por organismos públicos.

Para comenzar habrá que analizar primero el contenido de los diferentes tipos de certificado que emite Firmaprofesional. También hay que diferenciar el crear entidades finales del crear certificados de entidad final, son procesos diferentes pero a la vez están muy relacionados. En EJBCA el certificado de una entidad final se crea cuando se ha añadido correctamente dicha entidad. Para añadirla es necesario realizar el diseño de su certificado, esto implica configurar el perfil de certificado y el perfil de entidad final que corresponda a su política de certificación. Para dar de alta usuarios (y certificados) de prueba se pueden utilizar los perfiles ENDUSER y EMPTY, ambos son perfiles del tipo *FIXED* y de certificado y de entidad final respectivamente.

4.2.2.1 Contenido de los certificados

Los certificados de entidad final son los certificados que la CA subordinada de Firmaprofesional emite a las entidades finales. Según se analizó al principio de este trabajo Firmaprofesional emite diferentes tipos de certificado según la necesidad particular del cliente, las características de estos certificados están recogidas en sus respectivas políticas de certificación. Dichos certificados, así como los certificados de CA, cumplen con las recomendaciones *RFC 3280* y *RFC 3739* y son coherentes con lo dispuesto en las norma *ETSI TS 101 862*.

En la tabla de nombre *CONTENIDO DE LOS CERTIFICADOS RECONOCIDOS EMITIDOS POR FP* del anexo E se muestra el contenido de los campos básicos y de las extensiones de los diferentes tipos de certificado electrónico que emite la CA subordinada.

De esta tabla se ve que el *Subject DN* de los certificados de Persona Vinculada y de Factura Electrónica tienen los mismos campos, por lo que sólo será necesario crear un perfil de entidad final (ya que este perfil básicamente define el DN del sujeto). Este perfil será el utilizado a la hora de crear ambos tipos de certificado, esto se verá más adelante.

También se observa que algunos campos del *Subject DN* no son estándares, como el *1.3.6.1.4.1.4710.1.3.2* (del certificado de Persona Vinculada) y el *1.3.6.1.4.1.18838.1.1* (del certificado de Factura Electrónica), y que algunas extensiones X.509v3 tampoco lo son, como la *1.3.6.1.4.1.13177.10.1.5.1.1* y la *1.3.6.1.4.1.13177.10.1.5.1.2* (ambas del certificado de Persona Jurídica). Para más información acerca de estos campos y extensiones referirse al anexo A. El que no sean estándares significa que EJBCA no los conoce por lo que se les tendrá que añadir como campos de DN personalizados y como extensiones personalizadas respectivamente, el cómo hacerlo se verá en los puntos siguientes. Lo mismo se hará con la extensión *X509v3 Issuer Alternative Name* que a diferencia de las anteriores sí es estándar pero EJBCA no la implementa ni en esta versión ni en versiones superiores.

Como última anotación es importante resaltar que tanto la extensión *X509v3 Authority Information Access* como el campo de DN *Telephone Number*, a pesar de ser estándares, no están soportados en la versión 3.5.2 de EJBCA pero se ha visto que sí se implementan en versiones superiores del software, por esta razón no se añadirán a EJBCA hasta que se actualice la versión actual del software. El cómo realizar la actualización se explicará en un punto más adelante.

4.2.2.2 Añadir campos al DN

En EJBCA es posible añadir campos de DN (*Custom DN fields*) a los que ya se tienen disponibles para definir un DN, estos campos tienen que estar identificados por un OID privado o público el cual se añade a un fichero de configuración especial. Estos nuevos campos siempre serán codificados en UTF8 en el DN y se podrán añadir cuando se configure el *Subject DN* de los perfiles de entidad final y cuando se creen autoridades certificadoras (para esto hay que escribir el propio OID seguido de su valor en el Subject DN de la nueva CA).

El procedimiento a seguir es sencillo e implica la modificación de dos ficheros importantes:

- El fichero *ejbca_home/src/java/profilemappings.properties*, al final de este fichero habrá que añadir cada nuevo campo en una línea siguiendo el formato *"DN;DNName;DnId;ProfileName;ProfileId,ErrorString,LanguageConstant",* este fichero controla los campos disponibles para el DN de un sujeto en EJBCA. *DNName* y *ProfileName* tendrán el mismo valor que será el OID del nuevo campo, *DnId* y *ProfileId* también tendrán el mismo valor que será un número identificador único mayor a 100, *ErrorString* tendrá un valor que haga referencia al nuevo campo en casos de error y *LanguageConstant* será una constante, definida en los ficheros de idiomas, que contiene el nombre del campo.

- El fichero *ejbca_home/src/adminweb/languages/languagefile.XX.properties*, representa los ficheros de idiomas, XX representa un idioma y sus valores más importantes son *es* y *en*, para español e inglés respectivamente. En estos ficheros se escriben los valores de las constantes que se usan en la admin web, cada fichero tiene la traducción de la constante para el idioma que representa. En estos ficheros se tienen que definir los *LanguageConstant* que se utilizarán, no es necesario que se siga un orden alfabético.

Como se comentó anteriormente se necesitarán añadir como campos de DN los OIDs privados *1.3.6.1.4.1.4710.1.3.2* y el *1.3.6.1.4.1.18838.1.1*. Se muestra los pedazos de código añadidos a cada fichero, de todas formas ambos ficheros se incluyen en el anexo D.

En el fichero *profilemappings.properties*

```
...
DN;1.3.6.1.4.1.4710.1.3.2;101;1.3.6.1.4.1.4710.1.3.2;101;NIF/CIF;NIF/CIF
DN;1.3.6.1.4.1.18838.1.1;102;1.3.6.1.4.1.18838.1.1;102;NIF;NIF
...
```

En el fichero *languagefile.es.properties*

```
...
NIF/CIF      =      1.3.6.1.4.1.4710.1.3.2, NIF/CIF
NIF          =      1.3.6.1.4.1.18838.1.1, NIF
...
```

Luego de haber realizado estas modificaciones habrá que hacer un *ant deploy* y luego reiniciar JBoss, luego de hacerlo los nuevos campos aparecerán como un campo más en la lista de campos de DN disponibles.

Como último apunte, habrá que cuidar que los OIDs de los nuevos campos añadidos no se vuelvan estándares más tarde, porque si sucede esto EJBCA podría añadirlos como campos por defecto y el que estén doblemente definidos podría causar incongruencias.

4.2.2.3 Añadir extensiones básicas

En EJBCA también es posible añadir las extensiones que se necesiten y que no sean soportadas por la herramienta, esto se puede conseguir añadiendo extensiones básicas (*Custom Basic Extensions*). Las extensiones básicas solo contienen un valor estático y su implementación es simple porque se utiliza la clase *BasicExtension.java*, ubicada en *src/java/org/ejbca/core/model/ca/certextensions/*.

El procedimiento a seguir es sencillo e implica la modificación del fichero *ejbca_home/src/java/certextensions.properties*, en él a cada extensión se le tendrán

que definir unos parámetros. Todas las extensiones deben tener un identificador que sea único y es importante que estén ordenadas numéricamente según el identificador que tengan. Los parámetros son los siguientes:

- **oid**: El número único identificador de la extensión, tiene estar comprendido entre 1 y 255.
- **classpath**: El classpath a la clase BasicExtension.java, tendrá siempre el valor org.ejbca.core.model.ca.certextensions.BasicCertificateExtension.
- **displayname**: El nombre que tendrá la extensión en la página de edición de perfiles de certificado.
- **used**: Si la extensión se usará o si estará deshabilitada. Se usa TRUE o FALSE.
- **translatable**: Si el *display name* se deberá traducir según el fichero de idiomas. Se usa TRUE o FALSE.
- **critical**: Si la extensión será crítica o no. Se usa TRUE o FALSE.
- **property.encoding**: La codificación que se usará para el valor de la extensión. Siempre será DERUTF8STRING.
- **property.value**: El valor estático que tendrá esta extensión.

Luego de haber realizado estas modificaciones habrá que hacer un *ant deploy* y luego reiniciar JBoss, luego de hacerlo en la página de edición del perfil de certificado aparecerá una opción de nombre *"Used Custom Certificate Extensions"* que tendrá una lista con las extensiones agregadas.

En el fichero *certextensions.properties* (que se incluye en el anexo D) se observan las extensiones que se crearon, estas son la *X509v3 Issuer Alternative Name* y las no estándares *1.3.6.1.4.1.13177.10.1.5.1.1* y *1.3.6.1.4.1.13177.10.1.5.1.2.* Para añadirlas se deben editar los perfiles de certificado correspondientes y seleccionar las extensiones que necesitan. La primera la añadirán todos los perfiles de certificado de entidad final y el perfil *CAsub-sw*, y las otras dos las añadirán los perfiles de certificado de Persona Jurídica.

4.2.2.4 Configuración de los perfiles de certificado

Será necesario crear y configurar un perfil de certificado por cada política de certificación de Firmaprofesional. Es importante lograr que cada perfil tenga las extensiones especificadas en su respectiva política. Para más información acerca de cuales son estos campos y extensiones referirse al anexo A.

Para crear los certificados reconocidos para firma electrónica que ofrece Firmaprofesional se crearon en total once perfiles de certificado. En la siguiente tabla se observan los nombres de cada perfil creado y la política a la que corresponden, se ve que las políticas se diferencian entre las que guardan las claves en tokens hardware (en un *Dispositivo Seguro de Creación de Firma, DSCF*) y las que las guardan en tokens software (ficheros *P12*).

PERFILES DE CERTIFICADO SEGÚN POLÍTICAS			
	OID de la política	Política de Certificación	Nombre del Perfil de Certificado
En Dispositivo Seguro de Creación de Firma (DSCF)	1.3.6.1.4.1.13177.10.1.1.1	CP de Colegiado	colegiado-hw
	1.3.6.1.4.1.13177.10.1.2.1	CP de Persona Vinculada	vinculada-hw
	1.3.6.1.4.1.13177.10.1.5.1	CP de Persona Jurídica	juridica-hw, juridica_man-hw
	1.3.6.1.4.1.13177.10.1.6.1	CP de Factura Electrónica	factura-hw
	1.3.6.1.4.1.13177.10.1.9.1	CP de Firma Móvil	firmamovil-hw
En software	1.3.6.1.4.1.13177.10.1.1.2	CP de Colegiado	colegiado-sw
	1.3.6.1.4.1.13177.10.1.2.2	CP de Persona Vinculada	vinculada-sw
	1.3.6.1.4.1.13177.10.1.5.2	CP de Persona Jurídica	juridica-sw, juridica_man-sw
	1.3.6.1.4.1.13177.10.1.6.2	CP de Factura Electrónica	factura-sw

En el anexo E, en la tabla de nombre *CONFIGURACIÓN DE LOS PERFILES DE CERTIFICADO DE LAS ENTIDADES FINALES* se muestra la configuración de los perfiles de certificado que corresponden a los certificados reconocidos para firma electrónica, en concreto los valores que se le dieron a las opciones de configuración más importantes de cada perfil creado, no se muestran las opciones que se dejaron con su valor por defecto.

De esta tabla se observa que todos los perfiles de certificado tienen una validez de tres años (1095 días), porque así está escrito en sus políticas de certificación, menos el perfil de Firma Móvil ya que la validez de estos certificados la define el operador de Telecomunicaciones según su criterio, además este perfil tiene habilitada la opción que permite sobrescribir esta validez.

También se observa que los perfiles *juridica_man-hw* y *juridica_man-sw* solo se diferencian de los perfiles juridica-hw y juridica-sw en que los primeros utilizan la extensión *1.3.6.1.4.1.13177.10.1.5.1.1* y los segundos no, esta extensión es la que se usa para definir a las personas jurídicas mancomunadas.

4.2.2.5 Configuración de los perfiles de entidad final

Para completar el diseño del certificado de cada política de certificación será necesario crear y configurar perfiles de entidad final. En esta configuración cada uno de estos perfiles se asocia a un perfil de certificado (para que se pueda emitir el certificado según la política correspondiente), además es importante lograr que cada perfil de entidad final tenga los campos del *Subject DN* especificados en la política de certificación que define el perfil de certificado.

Con este objetivo se crearon cuatro perfiles de entidad final que completan el diseño de los certificados reconocidos para firma electrónica de Firmaprofesional. En la siguiente tabla se muestra la relación entre los perfiles creados, es decir el nombre de cada perfil de entidad final creado, asociado al nombre del perfil (o perfiles) de certificado que le corresponde. Se observa que el perfil EE-vinculada está asociado a los perfiles que definen los certificados de Persona Vinculada y los de Factura electrónica (tanto hardware como software), esto es porque estos dos tipos de certificado tienen el mismo *Subject DN* por lo que pueden utilizar el mismo certificado de entidad final.

RELACIÓN ENTRE PERFILES CREADOS	
Nombre del Perfil de Entidad Final	Nombre de los Perfiles de Certificado asociados
EE-colegiado	colegiado-hw, colegiado-sw
EE-vinculada	vinculada-hw, vinculada-sw, factura-hw, factura-sw
EE-juridica	juridica-hw, juridica-sw, juridica_man-hw, juridica_man-sw
EE-firmamovil	firmamovil-hw

En el anexo E, en la tabla de nombre *CONFIGURACIÓN DE LOS PERFILES DE ENTIDAD FIINAL* se muestra la configuración realizada para cada perfil de entidad final, en concreto los valores que se le dieron a las opciones de configuración de cada perfil creado.

Hay que resaltar que los perfiles *EE-vinculada*, *EE-factura* y *EE-jurídica* utilizan los campos de DN creados anteriormente para definir correctamente su *Subject DN*. También que no ha sido necesario definir atributos para el *Subject Directory Attribute Fields*, ni se utilizan las notificaciones vía mail ni la impresión de datos de usuario.

4.2.2.6 Añadir entidades finales

La adición de entidades finales está muy ligada a los perfiles de certificado y a los perfiles de entidad final, cuando se añade una entidad final se da de alta a un usuario de la CA, se le crea el par de claves criptográficas y el certificado de clave pública respectivo. Esto se realiza en la página *"Add End Entity"* del apartado *"RA functions"* del admin web y es tan sencillo como rellenar los campos que hay en esa página.

El componente RA de EJBCA es el que añade (también podría ser una RA externa) las entidades finales y para esto utiliza alguno de los perfiles de entidad final que se han creado, a su vez el perfil de entidad final le permite escoger el perfil de certificado que quiera utilizar. De forma sencilla se puede decir que el perfil de entidad final define la estructura del nombre propio de la entidad final y que el perfil de certificado define la estructura de las extensiones que tendrá su certificado.

En el anexo B se incluye en una figura la apariencia de la página de adición de entidades finales. Los campos que se muestran en esta página dependen de la configuración del perfil de entidad final, especialmente de si se han habilitado o no las casillas *Use*, *Default*, *Required* y *Modifiable* que tienen algunas opciones de configuración de este perfil.

En total se añadieron once entidades finales, una por cada perfil de certificado creado. Los certificados electrónicos de cada entidad posteriormente servirán para examinar si cumplen todo lo establecido en las políticas de certificación de Firmaprofesional, también se utilizarán para importarlos en la simulación del proceso de migración explicado más adelante. Estos certificados de prueba se incluyen en el anexo D, como resultados de este proyecto.

En la siguiente tabla se listan las entidades finales que se crearon, para cada una se especifica qué perfil de certificado se utilizó, qué perfil de entidad final se utilizó y a qué política de certificación pertenece la entidad final.

	ENTIDADES FINALES CREADAS			
	Nombre de la entidad final	**Perfil de Certificado**	**Perfil de Entidad Final**	**Política de Certificación de Firmaprofesional**
En Dispositivo Seguro de Creación de Firma (DSCF)	**usuario_colegiado_hw**	colegiado-hw	EE-colegiado	CP de Colegiado
	usuario_vinculada_hw	vinculada-hw	EE-vinculada	CP de Persona Vinculada
	usuario_juridica_hw	juridica-hw	EE-juridica	CP de Persona Jurídica
	usuario_juridica_man_hw	juridica_man-hw	EE-juridica	CP de Persona Jurídica
	usuario_factura_hw	factura-hw	EE-vinculada	CP de Factura Electrónica
	usuario_firmamovil_hw	firmamovil-hw	EE-firmamovil	CP de Firma Móvil
En software	**usuario_colegiado_sw**	colegiado-sw	EE-colegiado	CP de Colegiado
	usuario_vinculada_sw	vinculada-sw	EE-vinculada	CP de Persona Vinculada
	usuario_juridica_sw	juridica-sw	EE-juridica	CP de Persona Jurídica
	usuario_juridica_man_sw	juridica_man-sw	EE-juridica	CP de Persona Jurídica
	usuario_factura_sw	factura-sw	EE-vinculada	CP de Factura Electrónica

4.2.3 Generación de CRLs

Las listas de revocación de certificados (CRLs) de Firmaprofesional siguen lo recomendado en el estándar *RFC 3280* y son firmadas por la autoridad que ha emitido el certificado. Estas también tienen campos básicos y extensiones X.509v3 y, como en el caso de los certificados electrónicos, a la estructura de su contenido se le suele llamar perfil de CRL.

Que el perfil de las listas de revocación de Firmaprofesional sea idéntico al de las listas de EJBCA no es una tarea crítica de cara a los objetivos de este proyecto, solo basta con que las listas contengan la información de los certificados revocados por la CA.

En la tabla de nombre *CONTENIDO DE LAS CRLs DE FP Y DE EJBCA* del anexo E se muestran los campos y extensiones de las CRLS que emite la CA subordinada de Firmaprofesional y los que emite la *FP_CA1* creada con EJBCA.

Lo que sí es importante es que las CRLs se generen automáticamente de manera periódica y adecuada. Es decir, que cuando la fecha de próxima actualización de la CRL llegue exista un mecanismo que genere la nueva lista y que esta recoja correctamente la información de los certificados que últimamente se han revocado.

La configuración de la generación de las CRL en EJBCA se realiza al mismo tiempo que se crea la CA. En la parte *"CRL Specific Data"* de la página de creación de CAs se configuran las opciones que definen el perfil de CRL de la CA, el significado y la configuración de estas opciones ya se han explicado anteriormente.

Luego de esto, se ha de configurar un servicio de actualización de CRLs del tipo *"CRL updater"*, este servicio se encargará de actualizar las listas de revocación de las CAs añadiendo una entrada a la CRL por cada certificado revocado. Esto se hace en el enlace *"Edit Services"* del apartado *"System functions"*, en esta página se pueden crear, editar y borrar nuevos servicios, en general la página es bastante intuitiva. Se muestra en la siguiente tabla cómo se configuró este servicio.

CONFIGURACION DE SERVICIOS	
Opción	**Descripción**
Select Worker	Se elige el tipo de servicio que se creará, se pueden configurar servicios personalizados. Para este caso se eligió CRL UPDATER.
Select Interval	El tipo de intervalo en el que realizará su trabajo, se pueden configurar intervalos personalizados. Se eligió PERIODICAL INTERVAL.
Periodical Interval Settings:	
Period	Define en días/horas/minutos/segundos cada cuanto tiempo se ejecutará el servicio. Se configuró para que se ejecute cada 5 MINUTOS.
Select Action	Aquí se escoge la acción que se quiere que se realice cuando se ejecute el servicio, se pueden configurar acciones personalizadas. No es necesario definir ninguna acción.
Active	Se habilita esta casilla para que el servicio esté activo.

4.2.4 Los keystores de EJBCA

Los repositorios de claves o *keystores* son ficheros en los cuales se almacenan las claves privadas y los certificados de clave pública propios y los pertenecientes a entidades de confianza. Pueden estar en formato JKS o PKCS12.

Cuando se crea una nueva CA y se le quiere añadir como CA aceptable en la configuración de servidor SSL, o si se renueva el certificado de una CA, se puede instalar cualquier certificado de CA en el servidor SSL. Para esto primero se debe

añadir tal certificado al *keystore* de confianza (*java trust keystore)* de EJBCA *ejbca_home/p12/truststore.jks* y luego copiar este *keystore* en *jboss_home/server/default/conf/keystore/*. Estos pasos los realiza el comando:

```
ant –Dca.name=”Nueva_CA” javatruststore
```

Lo que escribe en la estándar output este comando es lo siguiente:

```
ejbca:javatruststore:
     [echo] Getting root certificate in DER format...
     [echo] ca getrootcert Nueva_CA /tmp/rootca.der -der
     [java] Wrote Root CA certificate to '/tmp/rootca.der'
     [echo] Adding to or creating keystore:
/usr/local/ejbca352_2/p12/truststore.jks
     [exec] error de keytool: java.lang.Exception: El alias <EJBCA-
CA> no existe
     [exec] Result: 1
     [exec] error de keytool: java.lang.Exception: El alias
<borrame2> no existe
     [exec] Result: 1
     [exec] Se ha añadido el certificado al almacén de claves
     [exec] [Almacenando /usr/local/ejbca352_2/p12/truststore.jks]
   [delete] Deleting: /tmp/rootca.der
     [echo] Merging available external modifications from
/usr/local/ejbca352_2/../ejbca-custom352.
     [copy] Copying 15 files to /usr/local/ejbca352_2

j2ee:deploytruststore:
     [copy] Copying 1 file to /usr/local/jboss-
4.2.2.GA/server/default/conf/keystore

BUILD SUCCESSFUL
Total time: 8 seconds
```

Luego de ejecutar este comando se debe reiniciar el servidor de aplicaciones JBoss. Hacer esto es importante porque *p12/truststore.jks* es el almacén de certificados de confianza que utilizan tanto el *JBoss web connector (tomcat)* como EJBCA, este almacén es el que contiene todas las CAs en las que confía para aceptar la autentificación con certificados de cliente en la admin web El servidor SSL envía una lista de las CAs aceptadas al cliente, y si en esta lista no aparece la CA que emitió su certificado no podrá seleccionar dicho certificado cuando intente acceder a la admin web.

Otro *keystore* importante es el que contiene el certificado SSL que usa JBoss para autentificar la admin web. En caso este certificado expirara y se tuviera que renovar o en caso se le quiera crear de nuevo (por ejemplo con un DN más adecuado), cuando ya se le tenga creado (o renovado, lo normal es que se cree con nombre *tomcat.jks*) se deberá copiar en *jboss_home/server/default/conf/keystore/* con nombre *keystore.jks*.

Se muestran los pasos para renovar el certificado SSL del usuario *tomcat* (es el nombre del usuario propietario del certificado) usando el CLI (siempre se utiliza desde *ejbca_home*), para más referencias acerca de los comandos y modo de empleo del CLI ver el anexo C.

```
Al usuario se le cambia el estado a New
bin/ejbca.sh ra setuserstatus tomcat 10

Se guarda el password en modo texto claro
bin/ejbca.sh ra setclearpwd tomcat <password de httpsserver.password>

Se genera el nuevo certificado, tendrá de nombre tomcat.jks
bin/ejbca.sh batch tomcat

Se copia el keystore al lugar correcto
cp p12/tomcat.jks $APPSRV_HOME/server/default/conf/keystore.jks
```

Luego de ejecutar estos comandos se habrá de reiniciar el servidor de aplicaciones JBoss.

A modo de resumen, se muestran los *keystores* que debería tener cada aplicación para que funcione correctamente.

- **EJBCA** (ejbca_home/p12): superadmin.p12, tomcat.jks, y truststore.jks. Los *keystores* de superadministrador, servidor SSL y el *java trust keystore* del servidor.
- **JBOSS** (*jboss_home/server/default/conf/keystore/)*: *truststore.jks* y *keystore.jks*. Ambos se copian desde p12/, al segundo se le cambia de nombre al copiarlo a JBoss (antes era *tomcat.jks*).
- **JAVA** (*java_home/jre/lib/security)*: *cacerts*. Este *keystore* no se utiliza casi para nada.

4.2.5 Actualización de la herramienta a una versión superior

En este punto se describen los procedimientos necesarios para actualizar la versión de la herramienta a una versión superior. Es importante mantener la herramienta actualizada porque una nueva versión puede incluir mejoras, corrección de errores y nuevas características que harán que permitirán un mejor funcionamiento de la herramienta.

Además, como se comentó anteriormente, la actualización también servirá para añadir a la configuración de los perfiles tanto la extensión *X509v3 Authority Information Access* como el campo de DN *Telephone Number*, ya que en la última versión disponible de la herramienta (la 3.7.1) sí están implementados.

4.2.5.1 Tipos de actualizaciones

Existen varias formas de actualizar la herramienta y cada una se aplica en un escenario concreto. Desde la versión 3.0 de EJBCA todas las actualizaciones que han habido se pueden diferenciar entre *major releases* y *minor releases*. Una *minor release* es una actualización que introduce mejoras a la versión anterior que no son lo suficientemente importantes como para que llegue a ser una *major release,* por ejemplo para la versión 3.5.2 una *minor release* puede ser la 3.5.3 y una *major* la versión 3.7.0. Cabe resaltar que una nueva actualización siempre incluye todas las mejoras de las actualizaciones precedentes.

Toda nueva versión de la herramienta tiene 3 ficheros relevantes que contienen los detalles de la versión, el fichero *ejbca_home/doc/RELEASE_NOTES* que tiene información de los cambios que aplica la versión y del tipo de actualización que es, el fichero *ejbca_home/Changelog.txt* que detalla de forma minuciosa estos cambios y el fichero *ejbca_home/doc/UPGRADE* que explica cómo realizar la actualización según la versión que se tenga.

Generalmente las actualizaciones dentro de una misma *major release* son *plug-in upgrades*, y como son *minor releases* los cambios que incluyen se pueden aplicar simplemente en tres pasos:

- Copiar el contenido de los directorios ejbca_home/conf/ y ejbca_home/p12/ de la versión anterior a la nueva.
- Inspeccionar el contenido del directorio ejbca_home/conf/ y agregar los cambios que hubieran de los ficheros *.properties.sample de la nueva versión a los *.properties de la antigua.
- Hacer un *ant deploy* con la nueva versión.

Se detallan los pasos que se siguieron para realizar la actualización de la versión 3.5.2 de la herramienta a la versión 3.7.1, este es un caso especial porque la nueva versión incluye cambios en la base de datos. Los dos primeros pasos son los mismos que para el caso de un *plug-in upgrade*.

```
Parar JBOSS y dentro del directorio ejbca_home ejecutar
ant bootstrap

Iniciar JBOSS (ignorar posibles errores) y desde ejbca_home
ant upgrade

El comando anterior realiza las modificaciones en la base de datos,
como se actualiza desde la versión 3.5.x habrá que responder NO a la
segunda y tercera pregunta.

Reiniciar JBOSS para borrar la caché
```

Si todo ha ido correctamente, al acceder a la admin web se comprobará que todas las configuraciones hechas durante este capítulo se mantienen.

4.2.5.2 El directorio ejbca-custom

A partir de la versión 3.5 de EJBCA se añadió la posibilidad de guardar las modificaciones y configuraciones personales de la herramienta en un directorio externo a *ejbca_home/*, este directorio se llama *ejbca-custom.* La idea es mantener todas estas modificaciones en un árbol aparte, la ubicación de este directorio se debe definir en el fichero *ejbca_home/conf/custom.properties* de la siguiente manera:

```
# Default location of customized changes to merge.
customejbca.home=${ejbca.home}/../ejbca-custom
```

Cada vez que se ejecute un comando *ant* se copiarán todos los ficheros que estén dentro de este directorio a *ejbca_home*, reemplazando los que pudieran existir en el mismo. Es importante que dentro de *ejbca-custom* se mantenga la misma estructura

de directorios que en *ejbca_home* y que se tenga en cuenta que luego de ejecutar *ant* no se podrán recuperar los ficheros reemplazados.

Este directorio es especialmente útil a la hora de realizar un *plug-in upgrade*, porque sólo habría que realizar el segundo y tercer paso de los explicados anteriormente. Hay que tener en cuenta que *ant deploy* será incapaz de verificar si las modificaciones que se han hecho son incompatibles con la nueva versión de EJBCA, por eso hay que tener especial cuidado (siempre mirar el fichero *UPGRADE*) cuando se quiera actualizar a una *major upgrade*.

4.2.5.3 Configuraciones finales

Si ya se tiene la herramienta actualizada a la versión 3.7.1 el último paso para terminar de configurar correctamente la *Jerarquía de Certificación de Firmaprofesional* es editar los perfiles, de certificado y de entidad final, correspondientes y añadirles tanto la extensión *X509v3 Authority Information Access* como el campo de DN *Telephone Number*. Una vez realizado esto, los perfiles creados en EJBCA estarán listos para emitir certificados que cumplan con las políticas de certificación de la empresa.

4.2.6 Limitaciones de los certificados de entidad final

Para analizar si existen limitaciones en los certificados que se han conseguido crear primero se debe saber como descargar los certificados de entidad final. Un administrador puede obtener el certificado cualquier entidad final desde la admin web, siguiendo el enlace *"List/Edit End Entities"* del apartado *"RA Functions"*. Este enlace lleva a una página en la que se pueden buscar entidades finales definiendo ciertos parámetros de búsqueda, cuando se encuentran las entidades deseadas se les puede editar la configuración y descargar su certificado electrónico. Los usuarios comunes de la herramienta pueden obtener el certificado de cualquier entidad final siguiendo el enlace *"Fetch User's lastest certificate"* de la public web, solo necesitarán escribir correctamente el DN completo de la entidad.

Examinando los certificados de entidad final que se crearon líneas atrás, se observa que todos cumplen lo establecido en las políticas de certificación de Firmaprofesional y esto es gracias a que se han logrado crear y agregar a la herramienta los campos de DN y extensiones no estándares descritos en ellas.

Sin embargo, aunque la extensión *1.3.6.1.4.1.13177.10.1.5.1.2* se ha logrado crear, no cumple óptimamente lo dispuesto en la política de los certificados de Persona Jurídica, ya que en ella se dice que en esta extensión se incluirán los *datos registrales* de la empresa a la que representa la persona jurídica tal y como aparecen en el registro mercantil. Esto quiere decir que esta extensión tendrá un valor distinto para cada persona jurídica a la que se le emita un certificado y en la forma en la que se ha creado, la extensión se tendrá que modificar para cada una de ellas. La utilización óptima de esta extensión implica que su valor se defina a la hora de añadir la entidad final, este desarrollo adicional no ha sido contemplado dentro de los objetivos de este proyecto.

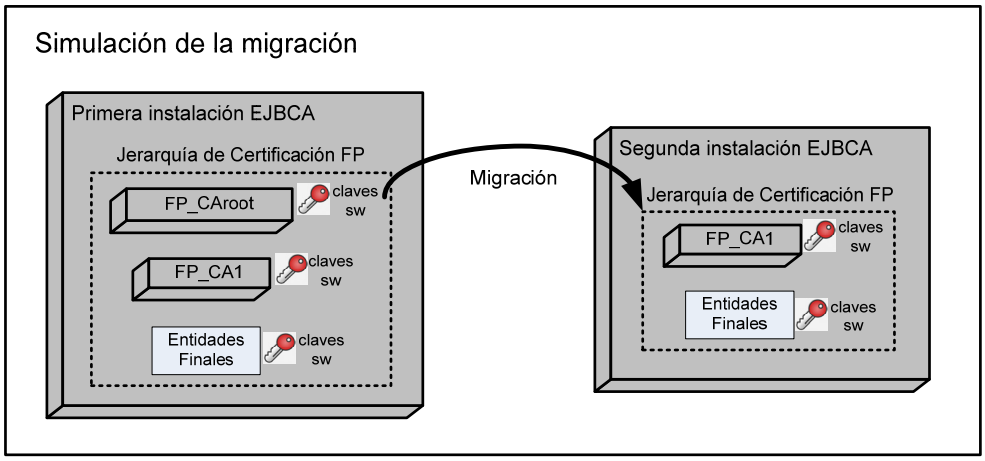
En la siguiente tabla se recopilan las principales limitaciones encontradas a la hora de crear los certificados reconocidos de entidad final en EJBCA y las medidas que se tomaron para resolverlas.

	LIMITACIONES DE LOS CERTIFICADOS DE ENTIDAD FINAL	
	Campo / Extensión	**Comentario**
Persona Jurídica	1.3.6.1.4.1377.10.1.5.1.1	Si la extensión aparece, su valor siempre será "Mancomunado". **(1)**
	1.3.6.1.4.1.18838.1.1	Este campo del Subject DN es utilizado solo por los certificados de Persona Jurídica. **(1)**
	1.3.6.1.4.1377.10.1.5.1.2	El valor de esta extensión serán los datos registrales de la persona jurídica y será diferente para cada persona jurídica. **(2)**
Otros certificados	X509v3 Authority Information Access	El componente CA Issuers de esta extensión, que utilizan todos los certificados de entidad final, está implementado en la versión 3.6.1 de EJBCA. **(3)**
	Telephone Number	Este campo del Subject DN del certificado de Firma Movil se implenta en la versión 3.7.1 de EJBCA. **(3)**
	1.3.6.1.4.1.4710.1.3.2	Este campo de Subject DN lo utilizan los certificados de Persona Jurídica y Factura Electrónica. **(1)**
	X509v3 Issuer Alternative Name	Esta extensión la utilizan todos los certificados y no es soportada en versiones superiores de EJBCA. **(1)**

(1) Se añadió como campo de Subject DN o como extensión básica.
(2) Se añadió como extensión básica. No funcionará óptimamente.
(3) Se añadió al actualizar la herramienta.

4.3 Simulación del proceso de migración

Luego de haber creado y configurado correctamente la *Jerarquía de Certificación de Firmaprofesional*, que era uno de los objetivos de este capítulo, habrá que simular el proceso de migración de toda esta jerarquía a una nueva instalación de la herramienta. Ésta, como se dijo al principio, es también una parte importante a desarrollar dentro del marco general de este proyecto porque de su resultado se evaluará si una futura migración, ya en el escenario real de la migración, será exitosa o no. A continuación se muestra una figura explicativa del proceso de migración.



En este apartado se detallan los principales pasos a seguir para lograr la migración de dicha jerarquía a la nueva instalación de la herramienta (versión 3.7.1), esta migración abarca la importación de las claves software de la autoridad *FP_CA1*, la importación de los perfiles previamente creados y por último la importación de todos los certificados de entidad final que emitió dicha autoridad.

En esta simulación la CA y las entidades finales que se usarán para evaluar la migración han sido creadas con el software de certificación EJBCA, en el escenario real estos elementos habrán sido creados por el antiguo software pero esta diferencia no tendría porqué representar una dificultad adicional a la hora de realizar la migración real.

4.3.1 Exportación e importación de CAs

Es recomendable que la CA sea el primer elemento de la jerarquía que se importe dentro de la nueva instalación, antes que los perfiles y que los certificados.

Solo será necesario importar a la nueva instalación la *FP_CA1*, pero para esto hay que lograr primero exportar sus claves software a un fichero P12 (un *keystore*). Este fichero además del par de claves criptográficas de la autoridad contendrá el certificado de la *FP_CA1*, de *FP_CAroot* y los demás certificados de confianza de la autoridad. A continuación se detallan los comandos de CLI para realizar la exportación y la importación de la CA.

```
Para exportarla desde la antigua instalación, se requerirá el
autentication code de la CA
bin/ejbca.sh ca exportca FP_CA1 fichero.p12

Para importarla en la nueva instalación, se requerirá el
autentication code de la CA
bin/ejbca.sh ca importca FP_CA1 fichero.p12 SignatureKeyAlias
EncryptionKeyAlias
```

En este caso las claves se exportarán al fichero *fichero.p12* y se usarán los alias por defecto *SignatureKeyAlias* y *EncryptionKeyAlias* como los alias de firma y encriptación respectivamente, si no se da un alias de encriptación EJBCA le creará a la CA un par de claves de encriptación.

En el escenario real de la migración se realizará la exportación e importación de la CA subordinada principal de Firmaprofesional (la *AC Firmaprofesional - CA1*), que tiene sus claves criptográficas dentro de un HSM y que fue creada con el antiguo software de certificación. Esta exportación requerirá de un procedimiento especial del HSM para recuperar esas claves y la importación utilizará un procedimiento especial de EJBCA para crear una CA a partir de un par de claves existentes en un HSM. El ahondar en estos procedimientos no fue contemplado dentro del desarrollo de este proyecto.

4.3.2 Exportación e importación de perfiles

Luego de tener la CA correctamente importada en la nueva instalación se procederá a importar todos los perfiles creados, tanto los perfiles de certificado como los de entidad final.

Primero se han de exportar los perfiles a un directorio previamente creado y luego se importarán a la nueva instalación desde ese directorio. A continuación se detallan los comandos de CLI para realizar la exportación y la importación de los perfiles.

```
Para exportarlos desde la antigua instalación
bin/ejbca.sh ca exportprofiles directorio_perfiles/

Para importarlos en la nueva instalación
bin/ejbca.sh ca importprofiles directorio_perfiles/ FP_CA1
```

El comando *exportprofiles* exportará en formato XML todos los perfiles existentes en la herramienta al directorio señalado, menos los *FIXED*, y les creará un nombre único a partir de sus nombres y de sus identificadores en la base de datos (tablas *EndEntityProfileData* y *CertificateProfileData*). Es importante conservar el identificador de cada perfil para mantener las referencias que existen entre los perfiles de certificado y los de entidad final. Los perfiles exportados son una parte importante de los resultados de este proyecto porque se utilizarán íntegramente en la futura migración de la empresa, están incluidos en el anexo D.

El comando *importprofiles* importará todos los perfiles que estén dentro del directorio señalado. Si a la hora de importar un perfil ya existiera otro perfil con el mismo nombre éste no sería importado, y si ya existiera otro perfil con el mismo identificador en la base de datos se trataría de escoger un nuevo identificador y de modificar las referencias. En la versión 3.7.1, si ninguna de las CAs configuradas como disponibles en los perfiles existe en la nueva instalación se tendrá que especificar una CA por defecto en el último argumento del comando. En este caso se eligió la *FP_CA1* pero en realidad no hacía falta hacerlo, esta opción hubiera sido útil en caso se hubieran importado los perfiles antes que la *FP_CA1*.

En el escenario real de la migración la exportación e importación de perfiles se realizará siguiendo estos mismos pasos, solo que a la hora de la importación se deberá escoger como CA por defecto a la *AC Firmaprofesional - CA1*.

4.3.3 Importación de los certificados de entidades finales

Por último se importarán las entidades finales que se hayan creado con la *FP_CA1*, esto implica también la importación de sus respectivos certificados. Esto es muy importante porque la autoridad necesitará, por razones obvias, reconocer como suyos los certificados que haya emitido en la antigua instalación, para poder así certificar que son confiables y evitar cualquier tipo de fraude, y el estado en el que se encuentran, para que sus CRLs incluyan los certificados revocados.

Para el caso de las entidades finales sí es imprescindible haber importado previamente la CA que las ha creado dado que una CA solo podrá importar los certificados que verdaderamente ha emitido. Además el sistema de seguridad de EJBCA garantiza que esto sea así verificando que el certificado de la entidad final que se quiere importar haya sido verdaderamente firmado por la CA.

A continuación se explica la sintaxis del comando de CLI para realizar la importación de las entidades finales, se tendrá que aplicar el mismo comando para cada entidad final que se importe. Vale la pena resaltar que para poder importar correctamente una entidad final se necesita como mínimo tener una copia en formato PEM de su certificado electrónico.

```
Para importar una entidad final
bin/ejbca.sh ca importcert <username> <password> FPCA1 <status>
      <certificate file> [<endentityprofile> | <endentityprofile>
      <certificateprofile>]
```

En <*username*> y <*password*> se debe escribir un nombre de usuario y una contraseña para la entidad que se añadirá. La tercera opción será siempre *FP_CA1,* porque es el nombre de la CA a la que se quieren importar los certificados de entidad final. En <*status*>, el estado que tiene el certificado de la entidad que se va a añadir, se debe escoger entre ACTIVE y REVOKED. En <*certificate file*>, la ruta del certificado en formato PEM de la entidad. La opción <*endentityprofile*> es opcional, se puede utilizar sola o junto con <*certificateprofile*>, estas opciones refieren al perfil de entidad final y al perfil de certificado con los que se quiere que sea añadido el certificado, estos perfiles deben existir y estar reconocidos por la CA, en este caso se escoge de entre los perfiles que se importaron según corresponda.

Por los motivos explicados anteriormente en el escenario real de la migración la importación de certificados es un tema delicado, por eso era importante probar en esta simulación si era posible realizarla. La importación de los certificados (en formato

PEM) de las once entidades finales, creadas cuando se crearon los certificados de prueba, se realizó con éxito por lo que se puede decir que esta importación en el escenario real no será un problema. En dicho escenario se tendrán que importar todos los certificados activos y revocados que ha emitido Firmaprofesional con su antiguo software de certificación, para hacerlo se utilizará este comando probablemente dentro de un *script* que automatice esta tarea. Como última anotación, a la hora de ejecutar el comando no será necesario utilizar las opciones <*certificateprofile*> ni <*endentityprofile*> y en el tercer campo se deberá especificar la *AC Firmaprofesional - CA1* como la autoridad a la que se importará los certificados.

Con este último paso se completó la importación de la *Jerarquía de Certificación de Firmaprofesional* creada en este capítulo, y vistos los resultados se puede decir que fue realizada correctamente.

CAPÍTULO 5: CONCLUSIONES

Luego del desarrollo del presente proyecto se puede decir que se han cumplido los objetivos inicialmente planteados. Los siguientes puntos recogen las conclusiones a las que se llegaron directamente tras cumplir con los objetivos iniciales.

- Se ha logrado determinar una secuencia de pasos inequívoca para la instalación de la herramienta, estos pasos incluyen algunas tareas previas a la propia instalación de la herramienta que son necesarias para que se logre instalar correctamente.

- Se ha logrado configurar en EJBCA la Jerarquía de Certificación de Firmaprofesional. En esta configuración ha sido importante conseguir que algunos campos de los certificados de las autoridades certificadoras creadas sean idénticos a los de los verdaderos, para que los certificados de entidad final que se emitan sean más parecidos a los que emite la empresa.

- En EJBCA los perfiles de certificado y de los perfiles de entidad final son los que definen el diseño de los certificados, en ellos se configuran exactamente los campos de DN y las extensiones que se quiere que tenga un certificado. Gracias a ellos se han podido configurar todos los tipos de certificados que emite Firmaprofesional.

- La simulación del proceso de migración permite tener una idea de lo que se realizará en la migración real de la empresa, pero en el escenario real de la migración aparecerán unas dificultades adicionales que no se resolvieron en este proyecto porque no se contemplaron dentro de los objetivos. Estas dificultades son la importación a EJBCA de una CA con claves almacenadas en un HSM y la integración del componente CA de la herramienta a la arquitectura de certificación de Firmaprofesional utilizando Web Services.

- La importación a EJBCA de una CA con claves software y de sus perfiles de certificado y entidad final es una tarea relativamente sencilla y se realiza con los comandos de la interfaz CLI: *importca*, *exportca*, *exportprofiles* e *importprofiles*.

- El comando *importcert*, de la interfaz CLI de EJBCA, permite importar los certificados digitales que se hayan emitido anteriormente con una autoridad de certificación siempre y cuando se haya importado previamente dicha autoridad. No es necesario que esta autoridad haya sido creada utilizando EJBCA.

Aparte de estas conclusiones, durante el desarrollo de este proyecto se obtuvieron otras conclusiones derivadas de los objetivos iniciales. Estas conclusiones son las siguientes:

- Las infraestructuras de clave pública son la base de los sistemas de seguridad modernos porque cuidan los siguientes aspectos de seguridad: Confidencialidad, Autenticación, Integridad y No repudio. Y, hoy en día, el salvaguardar estos aspectos es fundamental en una comunicación electrónica.

- Con la ley de firma electrónica que se introdujo en el 2003, cualquier empresa podrá desarrollar su actividad de negocio a través de Internet. Esta ley define la infraestructura para que tanto la empresa como el consumidor puedan

participar de la actividad comercial sin ningún riesgo. En este contexto los prestadores de servicios de certificación, específicamente sus autoridades de certificación, cumplen un rol importante.

- Hoy en día es posible construir una jerarquía de certificación utilizando soluciones opensource, ya que empiezan a ser igual de potentes que las de software privativo. Las grandes empresas, como Safelayer, que ofrecen soluciones de licencia privativa empiezan a tener en las soluciones opensource a un gran competidor.

- EJBCA, hoy por hoy, es la herramienta opensource más conveniente para construir desde una autoridad certificadora hasta una infraestructura de clave pública. Las principales razones que justifican esta afirmación son que no tiene coste de licencia, que es un proyecto continuado activo desde el 2001 que ofrece buena documentación y soporte técnico en las páginas relacionadas al proyecto, que ofrece múltiples funcionalidades y que en cada *major release* suma otras nuevas.

- Dentro de la jerarquía de certificación creada el que los certificados de las CAs creadas no sean exactamente iguales a los verdaderos no es una cuestión crítica, esto se debe a que estos certificados solo servirán para emitir certificados de entidad final que cumplan con los requerimientos de las políticas de certificación de la empresa.

- La creación de los certificados de entidad final también presentaron algunas limitaciones pero fueron resueltas de dos maneras: actualizando la herramienta a su última versión y personalizando la herramienta añadiéndole extensiones y campos de DN.

- Creando la Jerarquía de Certificación de Firmaprofesional y los perfiles de certificado y de entidad final necesarios, se demuestra que con EJBCA es posible emular la arquitectura de certificación de un PSC cualquiera sea el software de certificación que utilice su CA.

- Se ha facilitado el trabajo a Firmaprofesional de cara a su futura migración ya que se han obtenido resultados que serán aprovechados en ella, como los ficheros de configuración de la herramienta, los ficheros XML de los perfiles de certificado y de entidad final y la secuencia de pasos para lograr una correcta instalación del software.

Finalmente, se puede decir que Firmaprofesional, a partir de la evaluación de los resultados de este proyecto, está aun más convencida de la conveniencia de la migración del software de su CA a EJBCA, por lo que actualmente se encuentra desarrollando las fases previas al proceso de cambio de software.

CAPÍTULO 6: LINEAS FUTURAS

El trabajo desarrollado en el presente proyecto no es más que el paso previo a un proyecto más grande, el del cambio de software a EJBCA de la jerarquía de certificación de Firmaprofesional.

Para esto era necesario obtener algunos resultados, especificados en el anexo D, como los ficheros de configuración de EJBCA personalizados a las necesidades de la empresa y los ficheros XML que definen la configuración de los perfiles de certificado y entidad final creados. Dentro de estos resultados también se incluye la secuencia de pasos para lograr una correcta instalación del software, que no está dentro de dicho anexo.

Sin embargo, existen puntos concretos dentro de la migración real que no se trataron en este proyecto pero que serán importantes a la hora realizarla. Algunos de estos puntos implican tareas de investigación adicionales y, otros, desarrollos adicionales.

El primer punto es determinar los pasos necesarios para lograr importar a EJBCA una CA que tenga sus claves criptográficas almacenadas en un HSM, pudiendo así reutilizar las claves de la autoridad certificadora principal de Firmaprofesional, que será la que finalmente se importará.

El segundo, una vez importada la CA y cuando esté lista para emitir certificados, es lograr integrar el componente CA de la herramienta dentro de la arquitectura de certificación de Firmaprofesional. Exactamente esto significa desarrollar los procedimientos necesarios para que la CA de EJBCA se comunique con la RA de la arquitectura de la empresa, para esto se hará uso de la interfaz Web Services que ofrece EJBCA. Este punto implicará programar los ficheros XML que se encarguen de esta comunicación.

El último punto es optimizar el funcionamiento de la extensión *1.3.6.1.4.1.13177.10.1.5.1*, añadida como una extensión básica en la herramienta en los certificados de persona jurídica, de modo que el valor de esta extensión pueda cambiar de acuerdo a los datos de registro de cada persona jurídica a la que se le emita un certificado.

Firmaprofesional tiene pensado empezar a implementar los resultados anteriores y los que obtenga de los puntos que faltan desarrollar, primero, en un entorno de producción piloto para luego, si se ha comprobado que todo funciona correctamente, implementarlo en el entorno de producción real de la empresa. Una operación tan delicada y crítica como esta no se espera que se termine en un tiempo corto, podrá tomar algunos meses debido a que se realizará gradualmente y siguiendo el calendario de etapas que ha definido la empresa.

GLOSARIO DE TÉRMINOS

- **Active Directory**: Es la implementación de Microsoft del protocolo LDAP.
- **Admin web**: La interfaz gráfica de administración de la herramienta EJBCA.
- **API**: Acrónimo de *Application Program Interface*. Un API es un conjunto de comandos, funciones y protocolos que utilizan los programadores para construir un software para un sistema operativo específico.
- **CA**: Acrónimo de *Certificate Authority* (Autoridad de Certificación).
- **Clustering**: Es una arquitectura de servidor que simula el multiprocesamiento interconectando 2 o más computadoras para compartir la carga de procesamiento de una aplicación.
- **CP**: Acrónimo de *Certificate Policy* (Política de Certificación).
- **CPS**: Acrónimo de *Certification Practices* (Prácticas de Certificación).
- **CRL**: Acrónimo de *Certificate Revocation List*. Las listas de revocación de certificados son las que utilizan las CAs para publicar qué certificados han sido revocados, las CAs las emiten cada cierto tempo.
- **DN**: Acrónimo de *Distinguished Name* (Nombre Distintivo). Es un atributo especial que define inequívocamente una entrada. Esta formado por los componentes: *CN*, *O*, *OU*, *C*, etc.
- **DSCF**: Acrónimo de Dispositivo Seguro de Creación de Firma.
- **ECDSA**: Acrónimo de *Elliptic Curve Digital Signature Algorithm*. Es un algoritmo perteneciente a la criptografía de curva elíptica, la cual es la criptografía de clave asimétrica del futuro.
- **EJB specification**: La especificación EJB define un API que forma parte del estándar de construcción de aplicaciones empresariales J2EE, provee de un modelo de componentes distribuido estándar al servidor.
- **EJBCA**: Acrónimo de *Enterprise Java Beans Certificate Authority*.
- **FP**: Acrónimo del prestador de servicios de certificación Firmaprofesional, S.A.
- **Hard Token**: Cuando el *token* es un dispositivo electrónico hardware recibe este nombre. Las tarjetas criptográficas son un ejemplo de ellos.
- **HSM**: Acrónimo de *Hardware Security Module*. Es un tipo de *hard token* pero que permite realizar funciones criptográficas bajo el control de un ordenador.
- **J2EE specification**: La especificación J2EE define una plataforma de programación para desarrollar aplicaciones empresariales en JAVA de arquitectura distribuida de N niveles.

- **Jerarquía de Certificación**: Conjunto de elementos que forman parte del proceso de certificación. Generalmente formado, como mínimo, por una CA raíz en el nivel superior de la jerarquía y una o varias CAs subordinadas vinculadas a la misma. Adicionalmente puede tener una serie de elementos que proporcionan servicios al proceso de certificación.

- **JBoss**: Es un servidor de aplicaciones J2EE de código abierto implementado en JAVA.

- **Keystore**: Es una base de datos usada por la clase *KeyStore* para guardar sus propias claves privadas y los certificados de calve pública recibidos de otra persona. Un fichero *keystore* puede estar en formato PKCS12 o JKS.

- **LDAP**: Acrónimo de *Lightweight Directory Access Protocol.* Es un protocolo que permite el acceso a un servicio de directorio ordenado y distribuido para almacenar y buscar información.

- **Logging**: Se le llama así al proceso de almacenar información acerca de los eventos que ocurren en una aplicación.

- **Major Release**: Una nueva versión de un determinado software con posibles mejoras incompatibles con la versión anterior.

- **Minor Release**: Una nueva versión de un determinado software con posibles mejoras compatibles con la versión anterior.

- **OCSP**: Acrónimo de *Online Certificate Status Protocol*. Es un protocolo de Internet usado para obtener el estado de un certificado X.509 en un determinado instante. Fue creado como alternativa a las CRLs.

- **OID**: Acrónimo de *Object Identifier* (Identificador de objeto). Firmaprofesional utiliza OIDs definidos en el estándar ITU-T X.660 | ISO/IEC 9834-1:2005 *"Procedures for the Operation of OSI Registration Authorities: General Procedures and ASN.1 Object Identifier tree top arcs"*.

- **Opensource**: Refiere en general a cualquier programa cuyo código fuente está disponible para cualquier uso o modificación que puedan realizar otros usuarios. Estos programas normalmente son desarrollados por grupos públicos de desarrollo, este término no necesariamente implican que no haya coste de licencia.

- **PIN**: Acrónimo de *Personal Identification Number*. Es un código conocido solo por el propietario del dispositivo que utiliza el código PIN como medida de seguridad.

- **PKCS 10**: Acrónimo de *Public Key Cryptography Standard number 10.* Define un formato estándar para las peticiones de firma de certificado.

- **PKCS 12**: Acrónimo de *Public Key Cryptography Standard number 12.* Define un formato estándar para almacenar claves privadas acompañadas de sus certificados digitales, protegidos por una contraseña.

- **PKI**: Acrónimo de *Public Key Infrastructure* (Infraestructura de Clave Pública).

- **PKIX**: Acrónimo de *Public Key Infrastructure X.509* (Infraestructura de Clave Pública basada en certificados X.509).

- **PSC**: Acrónimo de Prestador de Servicios de Certificación.
- **Public web**: La interfaz gráfica de acceso público a la herramienta EJBCA.
- **PUK**: Acrónimo de *Personal Unblocking Key*. Este código resetea el dispositivo en el caso de que el usuario haya olvidado el código PIN.
- **RA**: Acrónimo de *Registration Authority* (Autoridad de Registro).
- **RSA**: Acrónimo de *Rivest*, *Shamir* y *Adleman*, los nombres de sus inventores. Es el algoritmo más conocido de la criptografía de clave pública, se puede utilizar tanto para encriptación como para firma digital.
- **Soft Token**: Es una versión software de un *hard token*. Un ejemplo de estos dispositivos son los repositorios de claves P12.
- **Token**: Dentro de la criptografía, se le llama así al dispositivo electrónico que utiliza un usuario autorizado para autenticarse a un servicio computarizado. Este dispositivo almacena las claves criptográficas del usuario.
- **TSA**: Acrónimo de *Time Stamping Authority* (Autoridad de Sellado de Tiempo).
- **VA**: Acrónimo de *Validation Authority* (Autoridad de Validación).
- **Web Services**: Es el nombre de una técnica de computación en la que mediante el uso de XML se intercambian datos entre programas u ordenadores.

BIBLIOGRAFIA

[1] Schneier, B. (1996). *Applied Cryptography* (2da Edition). John Wiley & Sons.

[2] Stallings, W. (2006). *Cryptography and Network Security. Principles and practices* (4th Edition). Pearson Prentice Hall.

[3] Noticias Jurídicas, NJ. *Ley 59/2003, de 19 de diciembre, de firma electrónica*. Obtenido el 4 de Abril de 2008, de http://noticias.juridicas.com/base_datos/Admin/l59-2003.html

[4] CincoDias.com (2003, Diciembre). *La ley de firma electrónica nace para potenciar las transacciones en la Red*. Obtenido el 4 de Abril de 2008, de http://www.cincodias.com/articulo/empresas/Ley/Firma/Electronica/nace/potenciar/transacciones/Red/cdscdi/20031212cdscdiemp_26/Tes/

[5] IBM developerWorks (2001, Febrero). *Easing the PAIN. How PKI can reduce the risks associated with e-business transactions*. Obtenido el 20 de Setiembre de 2007, de http://www.ibm.com/developerworks/library/s-pain.html

[6] The Internet Engineering Task Force, IETF (2008, Abril). *Public-Key Infrastructure (X.509) (pkix)*. Obtenido el 2 de mayo de 2008, de http://www.ietf.org/html.charters/pkix-charter.html

[7] DNI electrónico, Ministerio del Interior. http://www.dnie.es/

[8] Firmaprofesional, http://www.firmaprofesional.com/

[9] Firmaprofesional S.A. (2007, Julio) *Prácticas y Políticas de certificación.* Obtenido el 22 de abril de 2008, de http://www.firmaprofesional.com/cps/CPS-CPs_FP_003.0.pdf

[10] Network Working Group (2002, Abril). *Request for Comments: 3280*. http://www.ietf.org/rfc/rfc3280.txt

[11] Network Working Group (2001, Agosto). *Request for Comments: 3161*. http://www.ietf.org/rfc/rfc3161.txt

[12] EJBCA home page, http://ejbca.org/index.html

[13] EJBCA User Guide, http://ejbca.sourceforge.net/manual.html

[14] EJBCA Wiki, http://wiki.ejbca.org/

[15] EJBCA Blog, http://blog.ejbca.org/

[16] EJBCA Administration Tutorials, http://wiki.ejbca.org/admintutorials

[17] PrimeKey Solutions. *EJBCA Training.* Obtenido el 23 de Abril de 2008, de http://download.primekey.se/documents/ejbca_training.pdf

[18] LogicaCMG (2006). *Introduction to EJBCA*. Obtenido el 23 de Abril de 2008, de http://itsecug.se/files/ejbca-introduction-2007-05-09-v12.pdf

[19] Sourceforge.net: EJBCA, http://sourceforge.net/projects/ejbca/

[20] PrimeKey Solutions, http://www.primekey.se/primekey/en.html

[21] PrimeKey Solutios Online Documentation, http://docs.primekey.se/documentation/en.html

[22] Safelayer, http://www.safelayer.com/

[23] The Apache Ant Project, http://ant.apache.org/

[24] Lago, Ramiro (2006, OCtubre). *Log4j*. Obtenido el 3 de Julio del 2008, de http://www.proactiva-calidad.com/java/herramientas/log4j/index.html

[25] The Legion of the BouncyCastle, http://www.bouncycastle.org/

[26] MySQL, http://www.mysql.com/

[27] JBoss.org, http://www.jboss.org/

[28] Sun Microsystems, http://java.sun.com/

[29] JBoss Wiki (2008, Marzo). *Using mod_jk 1.2.x with JBoss/Tomcat bundle and Apache2*. Obtenido el 14 de Marzo de 2008, de http://wiki.jboss.org/wiki/UsingMod_jk1.2WithJBoss

[30] Facultad de Comunicación, Universidad de Piura. *Guía para citas y referencias bilbiográficas*. Obtenido el 17 de Abril de 2008, de http://www4.ujaen.es/~emilioml/doctorado/guia_rapida_de_citas_apa.pdf