IBM Cúram Social Program Management

Guía de seguridad de Cúram

Versión 6.0.5

IBM Cúram Social Program Management

Guía de seguridad de Cúram

Versión 6.0.5

Nota

Antes de utilizar esta información y el producto al que hace referencia, lea la información que figura en el apartado "Avisos" en la página 63

Revisado: mayo de 2013

Esta edición se aplica a IBM Cúram Social Program Management v6.0 5 y a todos los releases subsiguientes a menos que se indique lo contrario en nuevas ediciones.

Materiales bajo licencia - Propiedad de IBM.

© Copyright IBM Corporation 2012, 2013.

© Cúram Software Limited. 2011. Reservados todos los derechos.

Contenido

© Copyright IBM Corp. 2012, 2013

Figuras

© Copyright IBM Corp. 2012, 2013

Tablas

© Copyright IBM Corp. 2012, 2013

Capítulo 1. Seguridad de Cúram

1.1 Finalidad

La finalidad de esta guía es describir los aspectos de seguridad que deben tenerse en cuenta al desarrollar y desplegar una aplicación empresarial de IBM Cúram Social Program Management. El término Seguridad se utiliza para describir muchas áreas distintas. Para la finalidad de este documento, se cubren las siguientes áreas: autenticación y autorización. Esta guía también describe los elementos protegibles de aplicaciones de IBM Cúram Social Program Management.

Esta guía está dividida en dos partes. La primera parte contiene una descripción general de las áreas de seguridad y las partes interesadas deberían leerla, tanto arquitecto técnicos como desarrolladores. La segunda parte proporciona instrucciones de desarrollo y ejemplos de seguridad de la aplicación.

1.2 Público al que va dirigida

Este documento va dirigido a dos públicos principales:

- Arquitectos técnicos que deben tener en cuenta la integración con otros sistemas en el momento del despliegue, por ejemplo, LDAP.
- Desarrolladores que deben tener en cuenta el tipo de aplicación que van a desarrollar, es decir, si es una aplicación para usuarios internos, externos o ambos.

Nota: Los usuarios internos son usuarios que existen en la tabla de base de datos de usuarios de Cúram. Forman parte de la organización y normalmente están ahí para gestionar reclamaciones de participantes. Los usuarios externos son todos los otros tipos de usuarios. Los usuarios externos no forman parte de la organización. Su acceso está limitado. Un ejemplo de usuario externo sería un proveedor que proporciona un servicio a la organización.

1.3 Visión general

La seguridad está incorporada en la infraestructura que apuntala el desarrollo de la aplicación de IBM Cúram Social Program Management. Da soporte a la autenticación de un usuario cuando inicia la sesión, además de soportar el proceso de autorización. El diseño de una aplicación no se completa sin primero tener en cuenta las implicaciones de proteger la aplicación ante cualquier acceso no autorizado a datos o funcionalidades sensibles. Por ello, la seguridad es una de las prioridades clave durante el desarrollo de la aplicación.

A continuación presentamos conceptos principales de la seguridad de IBM Cúram Social Program Management:

- Autenticación
- Autorización

También existe el concepto de seguridad basada en la ubicación. En el nivel de organización, la seguridad de ubicación limita el acceso de un usuario a la información de clientes y casos. La seguridad de datos de ubicación también se puede configurar de modo que se permita que un usuario acceda a otras ubicaciones que no sean la suya propia. Esta guía no detalla la seguridad basada en la ubicación. Debe consultar la publicación *Cúram Location Administration Guide* (Guía de la administración de ubicaciones de Cúram) para obtener más detalles sobre la seguridad basada en la ubicación.

© Copyright IBM Corp. 2012, 2013

1.4 Capítulos de esta guía

Esta guía se divide en dos partes. La parte 1 consta de los siguientes capítulos, que proporcionan una descripción general de alto nivel de la arquitectura de seguridad de IBM Cúram Social Program Management de las aplicaciones y del despliegue en servidores de aplicaciones:

Capítulo 2 Autenticación
Este capítulo describe la arquitectura de autenticación de IBM Cúram Social Program Management, ofreciendo una descripción detallada de cada área y de todos los puntos personalizables disponibles.

Capítulo 3 Autorización
Este capítulo describe cómo funciona la autorización y cómo se puede configurar para usuarios.

Capítulo 4 Criptografía en Cúram
En este capítulo se describe cómo IBM Cúram Social Program Management utiliza la criptografía para proteger las contraseñas.

Capítulo 5 Almacenamiento en memoria caché de datos de seguridad
Este capítulo describe la memoria caché de seguridad incorporada de Cúram y también describe la memoria caché del servidor de aplicaciones de IBM® WebSphere y cómo afecta a la autenticación de usuarios.

Capítulo 6 Seguridad para clientes alternativos
Este capítulo describe los nombres de usuario que deben existir en la base de datos de usuarios de Cúram para garantizar que se ejecuten satisfactoriamente procesos como el flujo de trabajo.

Capítulo 7 Aplicaciones de usuarios externos
Este capítulo describe por qué debe añadirse una aplicación de usuario externo y qué debe tenerse en cuenta para hacerlo.

Capítulo 8 Utilización de un inicio de sesión único
Este capítulo describe las propiedades del servidor de aplicaciones que deben tenerse en cuenta cuando se va a utilizar IBM Cúram Social Program Management en una solución de inicio de sesión único.

Capítulo 9 Otras consideraciones sobre la seguridad
Este capítulo proporciona una breve visión general de algunas consideraciones y prácticas de seguridad externas.

La parte 2 consiste en los siguientes capítulos, que proporcionan varias actividades de desarrollo e instrucciones para codificar y personalizar la seguridad:

Capítulo 10 Personalización de la autenticación
Este capítulo describe puntos de personalización y artefactos de desarrollo relevantes para la autenticación.

Capítulo 11 Personalización de la autorización
Este capítulo describe cómo implementar la autorización para IBM Cúram Social Program Management.

Capítulo 12 Personalización de la criptografía
Este capítulo describe cómo personalizar el modo en que Cúram utiliza la criptografía.

Capítulo 13 Personalización de aplicaciones de usuario externo
Este capítulo describe cómo desarrollar una aplicación de usuario externo.

Capítulo 2. Autenticación

2.1 Visión general

Este capítulo describe la autenticación de IBM Cúram Social Program Management. La autenticación es el proceso de determinar si un usuario es quien dice ser. La autenticación es necesaria cuando debe verificarse un usuario para que pueda acceder a un recurso seguro de un sistema.

La autenticación basada en formularios es donde el usuario se presenta con un formulario que permite introducir las credenciales de nombre de usuario y contraseña. Estas credenciales se comparan con las credenciales almacenadas en el sistema para este nombre de usuario y, si coinciden, el usuario se considera un usuario autenticado en el sistema. Por motivos de seguridad, la contraseña de autenticación de un usuario se almacena en el sistema en forma de resumen (digest).

El cliente web de IBM Cúram Social Program Management se configura para dar soporte a la autenticación basada en formularios, lo que significa que antes de que un usuario pueda acceder a cualquier contenido del cliente web, se le va a redireccionar a un formulario de inicio de sesión para que se autentique.

El proceso de autenticación implica la verificación del nombre de usuario y de la contraseña, y ello lo realiza de forma predeterminada un módulo de inicio de sesión JAAS (Java™ Authentication and Authorization Service). De forma predeterminada, HTTPS/SSL están activos en el cliente web para garantizar que la modalidad de autenticación de inicio de sesión basado en formulario sea seguro.

2.2 Autenticación

Las distintas modalidades de autenticación se pueden configurar (en función de los requisitos de autenticación) mediante el módulo de inicio de sesión JAAS de Cúram.

Se admiten las siguientes modalidades de autenticación:

- Autenticación predeterminada
- Autenticación sólo de identidad
- Autenticación de seguridad de acceso externo.

Cada una de estas modalidades se describe de forma detallada en las secciones que encontrará más abajo.

© Copyright IBM Corp. 2012, 2013

2.3 Arquitectura de autenticación

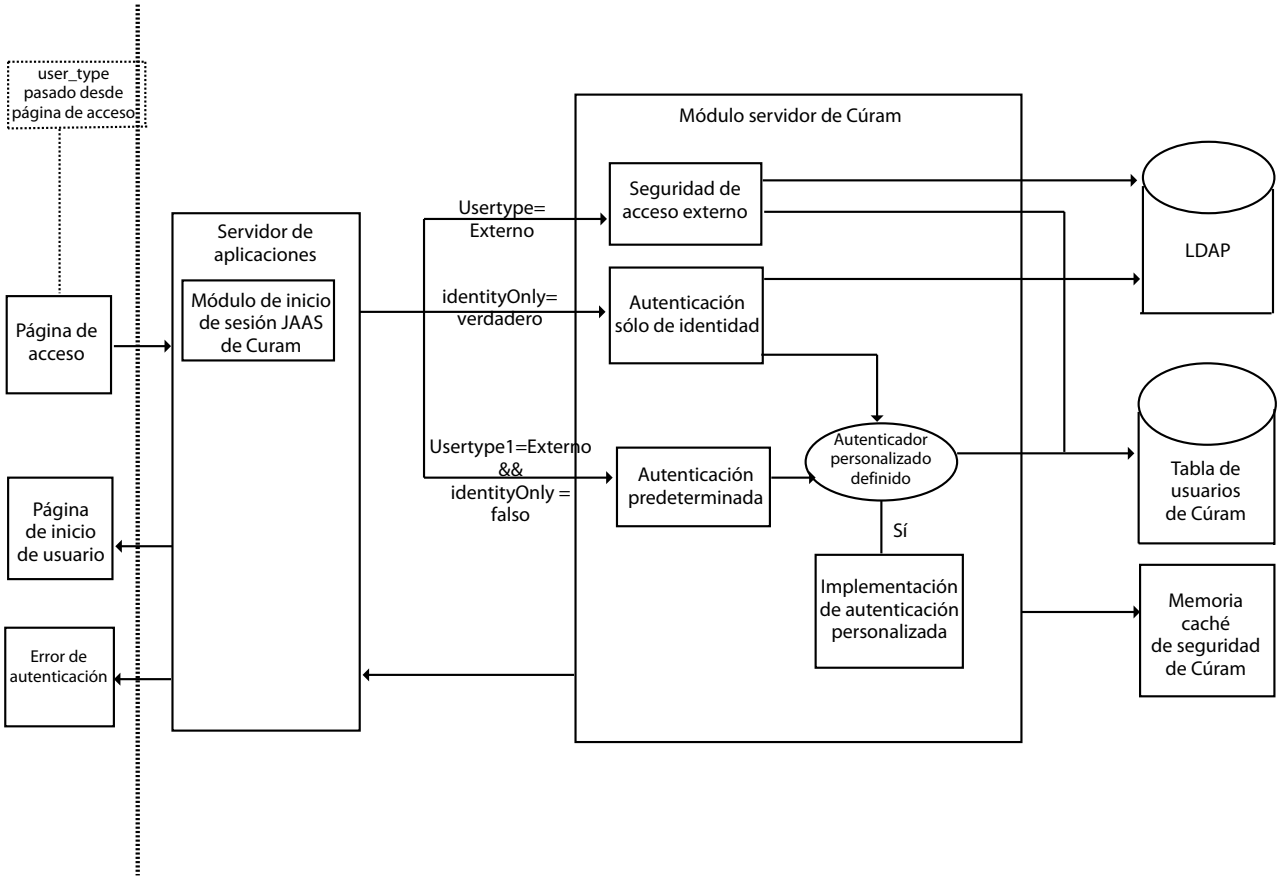

Figura 1. Arquitectura de autenticación

La anterior 2.3, "Arquitectura de autenticación" describe la arquitectura del proceso de autenticación de un usuario. Un usuario realiza la autenticación de forma rápida. Este comportamiento se puede personalizar tanto para los usuarios internos como los externos, dependiendo de los requisitos de autenticación. Las siguientes secciones de este capítulo describen cada una de las áreas funcionales que constituyen la arquitectura de autenticación, indicando dónde se pueden realizar personalizaciones.

2.4 Autenticación predeterminada

La autenticación instantánea predeterminada de IBM Cúram Social Program Management implica el inicio de sesión del usuario utilizando la pantalla de inicio de sesión, donde se solicita al usuario un nombre de usuario y una contraseña como credenciales. A continuación, estas credenciales se pasan al módulo de inicio de sesión JAAS de Cúram configurado en el servidor de aplicaciones.

Se invoca la autenticación predeterminada y el nombre de usuario y la contraseña introducidos se comprueban con el nombre de usuario y la contraseña almacenados en la base de datos de usuario de Cúram. El nombre de usuario de Cúram es inmutable, pero existe la opción de configurar el sistema para utilizar en su lugar un ID de inicio de sesión de Cúram, que sí es modificable. El ID de inicio de sesión es una extensión lógica del usuario Cúram, de modo que las verificaciones efectuadas con el nombre de usuario también se llevan a cabo con el ID de inicio de sesión. Consulte 2.5, "ID de inicio de sesión alternativos", en la página 5 para obtener información adicional relativa a los ID de inicio de sesión alternativos.

La autenticación realiza una serie de verificaciones contra las credenciales de inicio de sesión, consulte 2.14, “Autenticación predeterminada”, en la página 11 para obtener detalles sobre dichas verificaciones.

Si todas las verificaciones son satisfactorias, se considera que la aplicación ha autenticado el usuario.

Una vez el usuario se ha autenticado, dicho usuario se añade a la memoria caché de seguridad de Cúram. La memoria caché de seguridad de Cúram almacena el nombre de usuario y todos los datos de autorización relacionados con dicho usuario a fin de optimizar la recuperación de datos de autorización de un usuario. Debe consultar Capítulo 5, “Almacenamiento en memoria caché de datos de seguridad”, en la página 25 para obtener más detalles sobre la memoria caché de seguridad de Cúram. La figura 2.3 de más abajo resalta la vía de acceso tomada para la autenticación predeterminada.

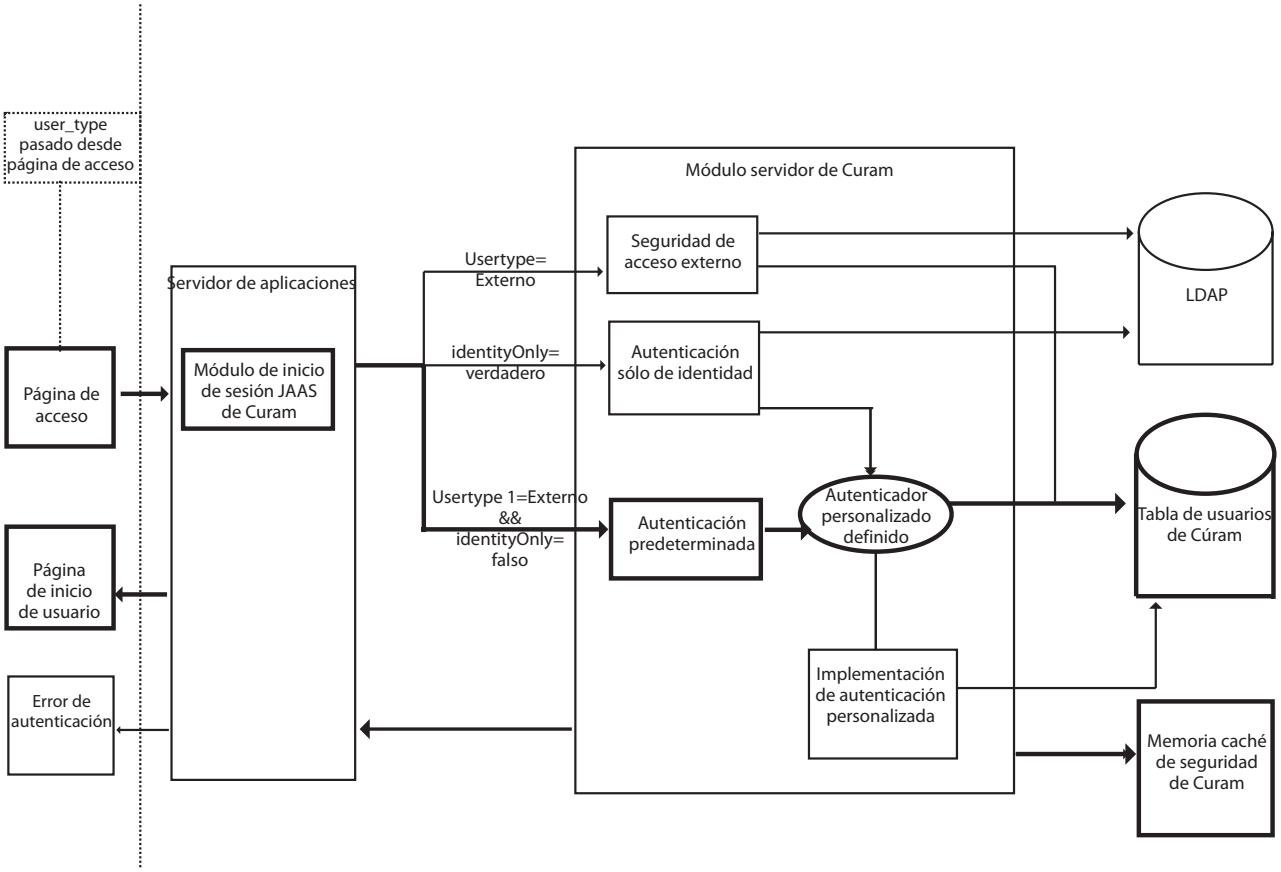

Figura 2. Autenticación predeterminada

2.5 ID de inicio de sesión alternativos

De forma predeterminada, Cúram utiliza el nombre de usuario y la contraseña resumida almacenados en la tabla Users para llevar a cabo la autenticación. Dicho nombre de usuario es inmutable, una vez creado no se puede cambiar. Es posible que esta falta de flexibilidad no se ajuste a las necesidades de determinadas instalaciones. En cualquier caso, existe la posibilidad de utilizar un ID de inicio de sesión, que puede actualizarse, en lugar del nombre de usuario inmutable. El ID de inicio de sesión funciona como una extensión lógica de la tabla Users de Cúram. Cuando se utiliza el ID de inicio de sesión alternativo, el nombre de usuario sigue existiendo y se utiliza internamente en Cúram, pero el usuario inicia sesión en Cúram utilizando el ID de inicio de sesión.

Cosas a tener en cuenta cuando se utiliza el ID de inicio de sesión:

- El ID de inicio de sesión y el nombre de usuario son de utilización mutuamente excluyente. Es decir, no se puede tener una mezcla de usuarios Cúram que inicien sesión unos con nombres de usuario y otros con ID de inicio de sesión.
- La tabla ExtendedUsersInfo de Cúram, donde se almacena el ID de inicio de sesión, debe rellenarse antes de activar la funcionalidad alternativa de ID de inicio de sesión, lo que se explica con más detalle a continuación.
- Cuando se utilizan ID, los resultados de las autenticaciones se almacenan en la tabla AuthenticationLog, y la columna AltLogin indica si la columna UserName representa un nombre de usuario (false) o un ID de inicio de sesión (true).
- Los ID de inicio de sesión solo son aplicables a usuarios internos de Cúram, es decir, usuarios almacenados en la tabla Users de Cúram. Sin embargo, si se está utilizando 'solo identidad' con los ID de inicio de sesión, e independientemente de donde estos estén almacenados (p.ej. en el registro de WebSphere, LDAP, etc.), deberán coincidir con los ID de inicio de sesión almacenados en la tabla ExtendedUsersInfo de Cúram.
- A la hora de asignar los ID de inicio de sesión, hay que tener cuidado con los ID utilizados internamente y/o que tengan dependencias (p.ej. con valores de propiedades) fuera de la tabla Users de Cúram. Estos son los nombres de usuario que darán problemas si su ID de inicio de sesión difiere del nombre de usuario sin el correspondiente cambio indicado:
  - SYSTEM - en WebSphere este nombre de usuario está asociado al procesamiento JMS y pasa a formar parte de la configuración de WebSphere en tiempo de despliegue de la aplicación. Consulte 6.2, "Usuarios de Cúram obligatorios", en la página 27 ya la correspondiente *Guía de despliegue de Cúram* en WebSphere para obtener información sobre cómo cambiar este ID.
  - DBTOJMS - este es el nombre de usuario predeterminado de DBtoJMS que utiliza el procesamiento por lotes y lo referencia la propiedad `curam.security.credentials.dbtojms.username`. Consulte 6.2, "Usuarios de Cúram obligatorios", en la página 27, 6.5, "Mensajería de JMS", en la página 28, 6.6, "Procesamiento aplazado", en la página 28 y la *Guía de procesamiento por lotes de Cúram* para obtener información adicional.
  - WEBSVCS - este es el nombre de usuario de servicios web predeterminado y lo referencia la propiedad `curam.security.credentials.dbtojms.username`. Consulte 6.2, "Usuarios de Cúram obligatorios", en la página 27, 6.3, "Servicios web", en la página 27 y la *Guía de servicios web de Cúram* para obtener información adicional.
  - unauthenticated - es el principal que utiliza WebSphere para los usuarios no autenticados; este ID de inicio de sesión no debe cambiarse.

Para habilitar el uso del ID de inicio de sesión alternativo, una vez rellenada la tabla ExtendedUsersInfo, establezca la propiedad `curam.security.altlogin.enabled` a `true` (consulte la *Guía del desarrollador de Cúram Server* para obtener información adicional sobre las propiedades de Curam). Se trata de una propiedad estática, por lo que deberá reiniciarse Cúram para que surta efecto.

Existen varias opciones para rellenar la tabla ExtendedUsersInfo antes de activar la funcionalidad; por ejemplo:

- Con una simple consulta SQL puede rellenarse la tabla utilizando el nombre de usuario de la tabla Users; esto no tiene un impacto directo sobre el usuario: `INSERT INTO EXTENDEDUSERSINFO (USERNAME, LOGINID, UPPERLOGINID, VERSIONNO) (SELECT USERNAME, USERNAME, UPPER(USERNAME), 1 FROM USERS);` Después podrán efectuarse las modificaciones a los ID de inicio de sesión de forma controlada.
- Puede desarrollarse una aplicación SQL que implemente la correlación de nombre de usuario e ID de inicio de sesión (p.ej. nombres comunes de LDAP).

  Nota: Deberá mantenerse la relación de clave foránea del nombre de usuario entre las tablas Users y ExtendedUsersInfo.

2.6 La página de inicio de sesión

La página de inicio de sesión instantáneo predeterminada viene representada por el archivo `logon.jsp`. Este archivo `logon.jsp` representa la página de inicio de sesión del usuario para que pueda completar la autenticación de inicio de sesión basado en formulario. De forma predeterminada, el archivo `logon.jsp` contiene los campos de nombre de usuario y de contraseña. No obstante, el archivo`logon.jsp` se puede personalizar para que pase un parámetro adicional añadiendo el campo user_type. Este campo determina el tipo de usuario que está iniciando la sesión, es decir, si se trata de un usuario interno o externo. El nombre de usuario, la contraseña y user_type (si existe) se pasan al módulo de inicio de sesión JAAS de Cúram como parte del proceso de autenticación.

El archivo `logon.jsp` inmediato predeterminado no tiene establecida la propiedad user_type. Si se omite esta propiedad, se asume que el usuario es interno. Cuando esta propiedad está establecida, indica que un usuario externo está iniciando la sesión. Esta propiedad se puede establecer con cualquier valor que no sea 'INTERNAL'.

2.7 Personalización de la página de inicio de sesión

El archivo `logon.jsp` se puede personalizar, es decir, el archivo `logon.jsp` se puede sustituir completamente por un archivo `logon.jsp` personalizado por numerosos motivos, incluidos los siguientes:

Se está desarrollando una aplicación cliente de usuario externo;
si se está desarrollando una aplicación cliente de usuario externo, debe crearse un nuevo archivo `logon.jsp` y debe establecerse el tipo de usuario para indicar que un usuario externo está iniciando la sesión. Debería consultar 13.3, “Creación de una página de inicio de sesión de usuario externo”, en la página 53 para obtener más detalles.

Se necesita un inicio de sesión automático;
algunas aplicaciones cliente de usuario externo no requieren autenticación del usuario y, por lo tanto, no debe solicitarse un nombre de usuario ni contraseña; es decir, en el caso de una aplicación de acceso público externo. No se puede inhabilitar la autenticación, por lo que la mejor forma de cumplir con este requisito es escribir un script de inicio de sesión automático. Puede hacerlo personalizando el archivo `logon.jsp` para la aplicación de acceso público externo. Debería consultar 13.4, “Creación de una página de inicio de sesión automática de cliente de usuario externo”, en la página 53 para obtener más detalles.

Se necesitan distintos estilos;
Debe consultar la sección de las páginas de inicio de sesión de la publicación *Cúram Web Client Reference Manual* (Manual de consulta de cliente web de Cúram) para obtener más detalles sobre los estilos del archivo `logon.jsp`.

Existe un requisito para que los nombres de usuario contengan caracteres ampliados (sólo válido para Oracle WebLogic Server).
WebLogic Server proporciona un atributo de propiedad, **j_character_encoding**, que debe añadirse al archivo `logon.jsp`. Consulte 10.3, “Habilitación de nombres de usuario con caracteres ampliados para WebLogic Server”, en la página 39 para obtener más detalles.

2.8 Módulo de módulo de inicio de sesión JAAS de Cúram

Un módulo de inicio de sesión JAAS realiza la autenticación. Se configura en el servidor de aplicaciones y es invocada automáticamente por el servidor de aplicaciones como parte del proceso de autenticación para todo acceso a la aplicación de IBM Cúram Social Program Management. La ventaja de este método es que el mecanismo de autenticación predeterminado se puede utilizar con (o ser sustituido por) un método personalizado sin que ello afecte a la aplicación de IBM Cúram Social Program Management.

Como se ha mencionado anteriormente, el módulo de inicio de sesión JAAS de Cúram se puede configurar para que funcione en tres modalidades. Para obtener más información sobre la configuración de los módulos de inicio de sesión en cualquier comportamiento específico del servidor de aplicaciones,

debe consultarse la sección de la Configuración del servidor de aplicaciones en *Cúram Server Deployment Guide* (Guía de despliegue del servidor de Cúram) del servidor de aplicaciones que se está utilizando para obtener más detalles.

Los requisitos específicos del proyecto pueden implicar que se necesita más de un módulo de inicio de sesión, es decir, el sistema puede solicitar al usuario que introduzca más información que el nombre de usuario y la contraseña para realizar la verificación. Se pueden configurar múltiples módulos de inicio de sesión en el servidor de aplicaciones. Cada módulo de inicio de sesión se ejecutará según el orden determinado en el servidor de aplicaciones. Para obtener más información sobre estos valores, consulte la documentación de WebSphere o WebLogic Server.

Una vez el usuario haya sido satisfactoriamente autenticado por todos los módulos de inicio de sesión que requieren una autenticación correcta del usuario (se puede configurar el en servidor de aplicaciones) el usuario ya se considera autenticado por la aplicación.

2.9 Gestión de contraseñas

Las contraseñas de todos los usuarios de Cúram internos y externos se almacenan en formato resumen en las tablas de base de datos Users y ExternalUsers de Cúram. Cuando el módulo de inicio de sesión JAAS de Cúram recibe la contraseña, se crea un resumen de la misma antes de enviarla al bean de inicio de sesión para su comparación. La creación de un resumen es un proceso unidireccional que garantiza la seguridad de la contraseña. La contraseña almacenada para el usuario en la base de datos utiliza el mismo algoritmo de creación de resúmenes, que está controlado por los valores criptográficos, por lo que se garantiza que las contraseñas encriptadas se puedan comparar satisfactoriamente las unas con las otras sin dejar de estar protegidas.

Los usuarios gestionados externamente como, por ejemplo, vía LDAP estando Cúram configurado con solo identidad, no están sujetos al proceso anterior. Cuando se autentica contra un sistema de terceros (p.ej. LDAP o un servidor de SSO) y la aplicación de Cúram necesita pasar las credenciales especificadas por el usuario al sistema de terceros, puede utilizarse la implementación personalizada de `curam.util.security.PublicAccessUser`, ya que permitirá acceder a las credenciales con una contraseña en claro.

2.10 Configuración predeterminada del servidor de WebLogic

El módulo de inicio de sesión JAAS de Cúram está configurado como un proveedor de autenticación en WebLogic Server. El proveedor de autenticación de Cúram es el único proveedor configurado por los scripts de configuración proporcionados para WebLogic Server. Puesto que es el único proveedor de autenticación configurado, el proveedor de autenticación de Cúram es responsable de autenticar y verificar el usuario. Como se ha mencionado anteriormente, es posible que haya más de un proveedor de autenticación configurado en WebLogic Server, en cuyo caso, es posible que el proveedor de autenticación de Cúram no sea el responsable de autenticar y verificar el usuario. Consulte 8.3, “Inicio de sesión único de WebLogic Server”, en la página 36 para obtener detalles adicionales.

2.11 Configuración predeterminada de WebSphere

El módulo de inicio de sesión JAAS de Cúram se configura como un módulo de inicio de sesión en el sistema en WebSphere. De forma predeterminada, la configuración de seguridad generada por script en WebSphere implica el registro de usuarios basado en archivos y el módulo de inicio de sesión en el sistema de Cúram. El registro de usuarios en WebSphere es el mecanismo de autenticación predeterminado y se puede configurar para que sea:

- un registro de usuarios personalizado
- un servidor de directorio LDAP;
- el SO local o
- el repositorio basado en archivos de WebSphere.

Hay múltiples configuraciones de inicio de sesión en el sistema de WebSphere. El módulo de inicio de sesión en el sistema de Cúram está configurado para las configuraciones `DEFAULT`, `WEB_INBOUND` y `RMI_INBOUND`. Se utiliza el mismo módulo de inicio de sesión para las tres configuraciones. WebSphere invoca automáticamente los módulos de inicio de sesión configurados como módulos de inicio de sesión en el sistema bajo determinadas circunstancias:

- `DEFAULT`

  Los módulos de inicio de sesión especificados para la configuración DEFAULT se invocan para la autenticación de servicios web e invocaciones JMS. También se invocan durante la fase de inicio de WebSphere ;
- `WEB_INBOUND`

  Los módulos de inicio de sesión especificados para la configuración WEB_INBOUND se utilizan para la autenticación de solicitudes web;
- `RMI_INBOUND`

  Los módulos de inicio de sesión especificados en la configuración RMI_INBOUND se utilizan para la autenticación de clientes Java.

El módulo de inicio de sesión JAAS de Cúram existe como un módulo de inicio de sesión dentro de una cadena de módulos de inicio de sesión configurada en WebSphere. Se espera que como mínimo uno de estos módulos de inicio de sesión sea el responsable de añadir las credenciales del usuario. De forma predeterminada, el módulo de inicio de sesión de Cúram añade las credenciales del usuario autenticado. En consecuencia, el registro de usuarios de WebSphere configurado manejado por un módulo de inicio de sesión subsiguiente no añade credenciales. Por ello, no es necesario definir usuarios de Cúram en el registro de usuarios de WebSphere. Este comportamiento se puede configurar utilizando la propiedad curam.security.user.registry.enabled establecida en el archivo `AppServer.properties`. Debe consultar la *Cúram Deployment Guide for WebSphere Application Server* (Guía de despliegue de Cúram de WebSphere Application Server) o la *Cúram Deployment Guide for WebSphere Application Server on z/OS* (Guía de despliegue de Cúram de WebSphere Application Server en z/OS) para obtener más detalles sobre cómo establecer esta propiedad. En 2.11, “Configuración predeterminada de WebSphere”, en la página 8 verá ilustrado el flujo de autenticación predeterminado de WebSphere. En 2.11, “Configuración predeterminada de WebSphere”, en la página 8 verá ilustrado el flujo de autenticación de WebSphere en el que también se necesita el registro de usuarios, es decir, donde la propiedad curam.security.user.registry.enabled está establecida a `true`.

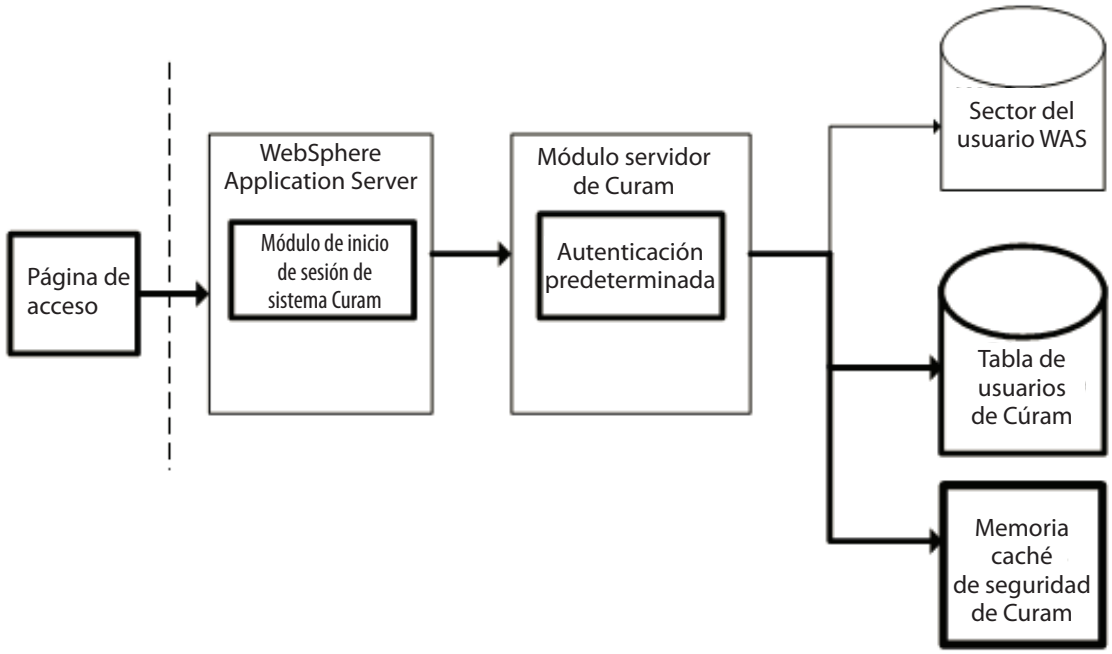

Figura 3. Flujo de autenticación predeterminado de WebSphere

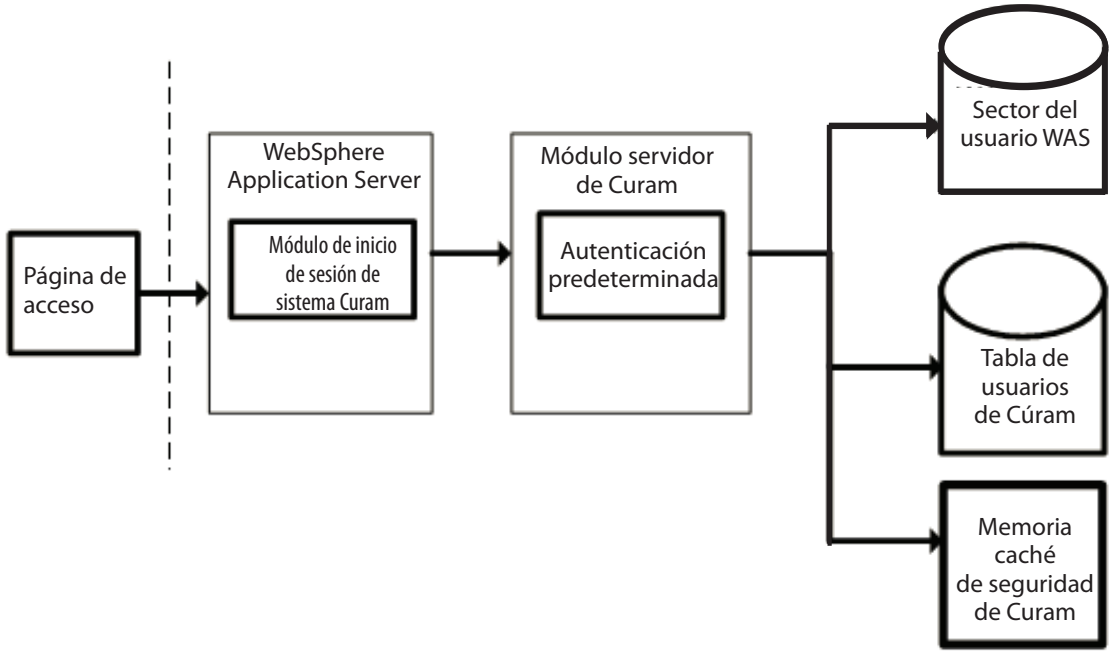

Figura 4. Flujo de autenticación de WebSphere con el registro de usuarios habilitado

Como parte de la configuración de seguridad, existen determinados usuarios que se excluyen de la autenticación y para dichos usuarios, el registro de usuarios configurado *será* necesario. Esta lista de usuarios se configura automáticamente para que sea el usuario de seguridad de WebSphere, tal como se especifica en la propiedad security.username en `AppServer.properties`, y el usuario de base de datos, tal como lo especifica la propiedad curam.db.username en `Bootstrap.properties`. Estos dos usuarios se clasifican como usuarios administrativos y no como usuarios de la aplicación. Se puede ampliar esta lista de usuarios excluidos de forma manual. Consulte las publicaciones *Cúram Deployment Guide for WebSphere Application Server* (Guía de despliegue de Cúram WebSphere Application Server) y *Cúram Deployment Guide for WebSphere Application Server on z/OS* (Guía de despliegue de Cúram de WebSphere Application Server en z/OS) para obtener más información.

Advertencia: Los scripts de configuración facilitados añaden automáticamente los usuarios security.username y curam.db.username al repositorio de usuarios basado en archivos de WebSphere. Si el registro de usuarios configurado de WebSphere no es el predeterminado, estos usuarios deben existir en el registro de usuarios alternativo de WebSphere.

2.12 Personalización del módulo de inicio de sesión JAAS

Es posible que el módulo de inicio de sesión JAAS de Cúram no soporte los requisitos de autenticación de una solución personalizada en concreto. Es muy recomendable que al desarrollar un módulo de inicio de sesión personalizado, se deje el módulo de inicio de sesión JAAS de Cúram en su lugar y se utilice con la autenticación sólo de identidad habilitada. No obstante, si se considera necesario, el módulo de inicio de sesión JAAS de Cúram se puede eliminar y se puede sustituir por una solución personalizada. Si así lo desea, debe ponerse en contacto con el servicio de soporte.

Advertencia: Aunque se puede eliminar el módulo de inicio de sesión JAAS de Cúram completamente, debe tenerse en cuenta que los usuarios deben seguir existiendo en la tabla de base de datos de usuarios de Cúram por razones de autorización.

El módulo de inicio de sesión JAAS de Cúram añade usuarios nuevos a la memoria caché de seguridad de Cúram automáticamente y cuando este módulo de inicio de sesión JAAS de Cúram es sustituido por un módulo de inicio de sesión JAAS personalizado, esta funcionalidad ya no está presente. Si un módulo de inicio de sesión JAAS personalizado va a sustituir completamente el módulo de inicio de sesión JAAS de Cúram, es responsabilidad del módulo de inicio de sesión JAAS personalizado el garantizar que se desencadene una actualización de la memoria caché de seguridad cuando se añada un usuario nuevo en la base de datos.

2.13 Proceso de verificación de autenticación

El tipo de verificaciones realizado depende de la modalidad de autenticación que se esté utilizando. A continuación encontrará una lista de las modalidades/configuraciones de autenticación y detalles completados sobre las verificaciones completadas para cada modalidad de autenticación.

2.14 Autenticación predeterminada

La autenticación predeterminada forma parte de la configuración instantánea y esta modalidad de autenticación implica la verificación del nombre de usuario y de la contraseña especificados durante el inicio de sesión frente a la tabla de base de datos de usuarios de Cúram. Toda la información de inicio de sesión en este caso es actualizada por la aplicación de IBM Cúram Social Program Management.

2.15 Proceso de verificación predeterminado

Las verificaciones efectuadas por el módulo de inicio de sesión de Cúram durante la autenticación predeterminada son:

- nombre de usuario y contraseña.
- caducidad de cuenta y/o contraseña.
- sincronización de nombre de usuario con memoria caché de seguridad.
- detección de intrusión, por ejemplo, límite superior de intentos de entrada de contraseña, nombres de usuario incorrectos, errores en cambio de contraseña.
- restricciones de acceso de día y hora; día de la semana y rango de tiempo del día.

La autenticación y la autorización de nombres de usuario es sensible a mayúsculas y minúsculas de forma predeterminada, no obstante, se puede inhabilitar la autenticación sensible a mayúsculas y minúsculas. Si existe un caso duplicado de nombres de usuario no sensibles a mayúsculas y minúsculas (por ejemplo, asistente social y Asistente Social), la autenticación fallará por tener un nombre de usuario ambiguo. Debe consultar 10.4, “Cómo cambiar la sensibilidad a mayúsculas y minúsculas del nombre de usuario”, en la página 39 para obtener más detalles sobre este tema.

2.16 Intentos de autenticación

Los errores de autenticación no se notifican directamente a un cliente puesto que éste proporcionaría información extra a cualquier intruso que quisiera acceder en el sistema. Por ejemplo, el hecho de notificar una contraseña incorrecta podría indicar que el nombre de usuario es válido. En su lugar, todos los intentos de autenticación (tanto correctos como incorrectos) se registran en una tabla de base de datos denominada AuthenticationLog . Debería consultar 11.15, “Análisis de la tabla de base de datos AuthorisationLog”, en la página 46 para obtener más detalles.

2.17 Personalización de la autenticación predeterminada

La implementación predeterminada se puede personalizar para que utilice un ID de inicio de sesión mutable en lugar del nombre de usuario de Curam, y la posibilidad de añadir verificaciones adicionales está disponible mediante la aplicación de un autenticador personalizado (consulte 2.21, “Verificaciones personalizadas”, en la página 15 para obtener detalles adicionales).

2.18 Autenticación sólo de identidad

La autenticación se puede configurar para realizar la verificación sólo de identidad, en lugar de las verificaciones predeterminadas listadas en2.15, “Proceso de verificación predeterminado” más arriba.

La verificación de sólo identidad significa que el mecanismo de autenticación sólo garantiza que el nombre de usuario del usuario que está iniciando la sesión existe en la tabla de base de datos de usuarios de Cúram. Un mecanismo alternativo, que debe configurarse en el servidor de aplicaciones, debe completar la autenticación completa.

Un ejemplo de mecanismo alternativo es un servidor de directorio LDAP, que es soportado como un mecanismo de autenticación por parte de los servidores de aplicaciones WebSphere y WebLogic Server. Otra alternativa es utilizar una solución de inicio de sesión único para la autenticación o implementar un módulo de inicio de sesión personalizado. Debe consultar la documentación de IBM u Oracle acerca de las soluciones del servidor de aplicaciones personalizado.

Con la autenticación sólo de identidad (como para la autenticación predeterminada), las entradas se añaden a la tabla de base de datos AuthenticationLog al final del proceso de autenticación.

Para conseguir un inicio de sesión satisfactorio, se utiliza el siguiente estado:

- AUTHONLY

Para un escenario anómalo, se utiliza el siguiente estado:

- BADUSER

  Se trata del único escenario de error posible en el que el usuario no existe.

Los campos loginFailures y lastLogin del AuthenticationLog no están establecidos. Ello sigue aplicándose incluso si se implementan verificaciones personalizadas.

Cuando se establece la información de caducidad de la contraseña de un usuario (en la tabla de base de datos de usuarios de Cúram), se visualiza una advertencia de caducidad de la contraseña si ésta va a caducar. Con la autenticación sólo de identidad esta advertencia es engañosa. Es recomendable que todos los campos relacionados con las verificaciones de autenticación, tales como la caducidad de contraseña o cuenta habilitada, no se utilicen si se ha habilitado la autenticación sólo de identidad.

Cuando la autenticación sólo de identidad está habilitada, la seguridad no se utiliza para la autenticación pero se sigue utilizando en miras a la autorización. En consecuencia, todos los usuarios que necesitan acceder a la aplicación deben seguir existiendo en la tabla de base de datos de usuarios de Cúram, además de en el mecanismo de autenticación alternativo, por ejemplo, LDAP. Es importante tener en cuenta que hay dos usuarios que deben existir en ambas ubicaciones, es decir, el usuario SYSTEM y el usuario DBTOJMS. Debe consultar Capítulo 6, “Seguridad para clientes alternativos”, en la página 27 para obtener más detalles sobre estos usuarios.

Debe consultar 10.7, “Configuración de la autenticación sólo de identidad”, en la página 40 para obtener detalles sobre cómo configurar "sólo de identidad" para un servidor de aplicaciones.

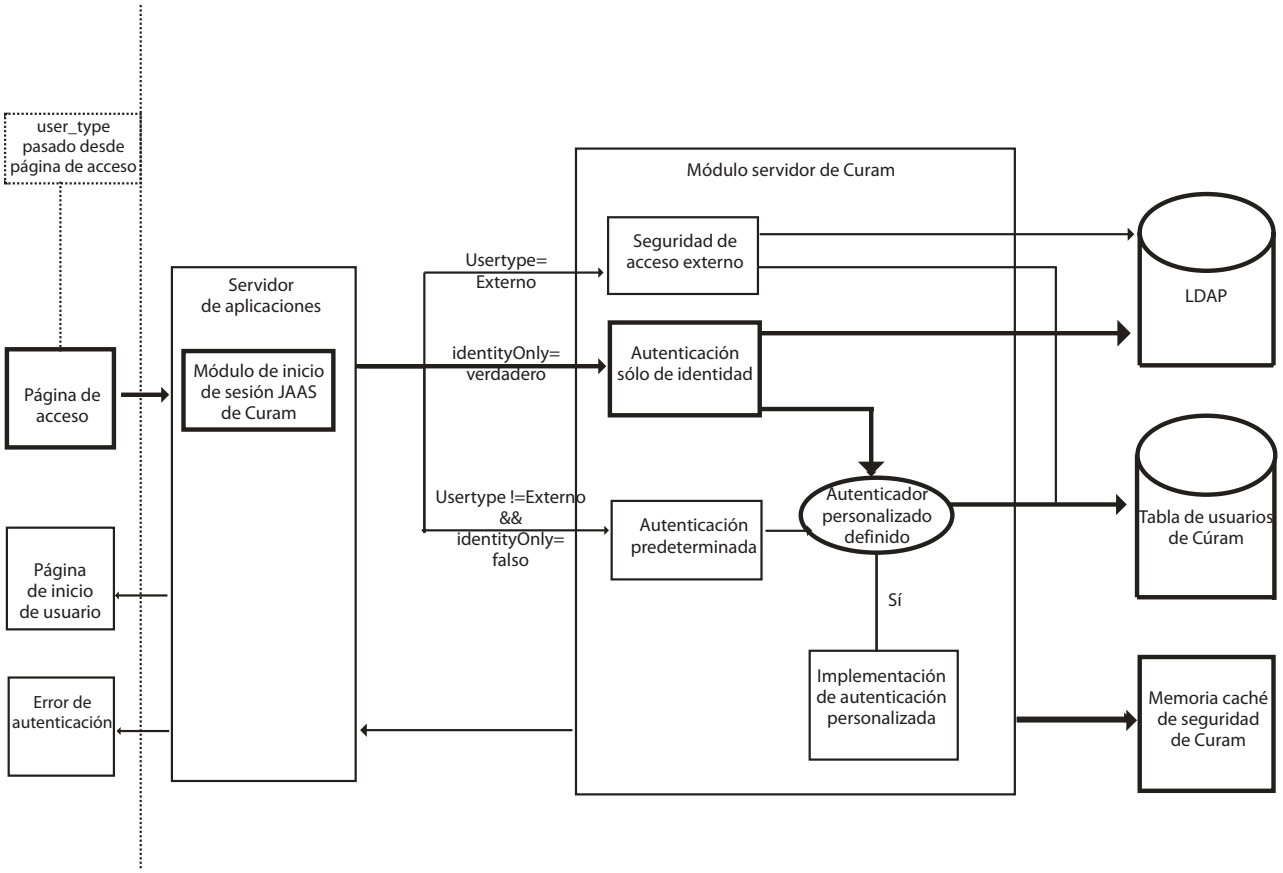

Figura 5. Autenticación sólo de identidad

2.19 Personalización de la autenticación sólo de identidad

La implementación sólo de identidad no se puede personalizar, pero se pueden añadir verificaciones extras implementando el autenticador personalizado. Debería consultar 2.21, “Verificaciones personalizadas”, en la página 15 para obtener más detalles.

2.20 Autenticación de seguridad de acceso externo

La arquitectura permite que un desarrollador implemente su propia solución de autenticación personalizada para usuarios externos proporcionando un “gancho” en la infraestructura de autenticación y autorización existente del SDEJ.

Para “enganchar” la solución personalizada a la aplicación, deberá extenderse la clase `curam.util.security.PublicAccessUser`, lo que requiere implementar la interfaz `curam.util.security.ExternalAccessSecurity`. Esta clase se utiliza durante el proceso de autorización y autenticación para determinar la información necesaria relacionada con el usuario externo. Consulte Capítulo 13, “Personalización de aplicaciones de usuario externo”, en la página 53 para obtener detalles adicionales.

2.21 Verificaciones personalizadas

Se proporciona soporte para añadir verificaciones personalizadas al proceso de autenticación, por ejemplo, es posible que se pida a un usuario que conteste una pregunta de seguridad que a continuación deba verificarse. El código personalizado, si se ha implementado, es invocado después de las verificaciones de IBM Cúram Social Program Management relevantes o la aserción de identidad, y sólo si han sido satisfactorias.

Después de invocar las verificaciones personalizadas, el proceso de autenticación actualiza los campos relevantes en la tabla de base de datos de usuarios.

Debería consultar 10.5, “Cómo añadir verificaciones personalizadas al proceso de autenticación”, en la página 40 para obtener más detalles.

Capítulo 3. Autorización

3.1 Visión general

En IBM Cúram Social Program Management, el proceso de otorgar o rechazar un acceso de usuario a elementos funcionales de una aplicación se llama autorización. El elemento funcional puede ser cualquier cosa a la que se puede adjuntar un identificador exclusivo, como por ejemplo:

- una llamada de proceso de servidor,
- un elemento de la aplicación que requiere comprobación de seguridad, por ejemplo, una serie de productos para el bienestar registrados.

El acceso al elemento funcional está controlado por un identificador de seguridad (SID) que forma parte de los datos de autorización de IBM Cúram Social Program Management. Estos datos están enlazados con un usuario y se pueden configurar mediante las pantallas de administración de Cúram o mediante Data Manager. Debe consultar la publicación*Cúram Server Developer's Guide* (Guía del desarrollador del servidor de Cúram) para obtener más detalles.

Los datos de seguridad creados para la autorización son fundamentales para el proceso realizado durante cada llamada cliente-servidor, y es importante que se optimice el acceso por razones de rendimiento. La memoria caché de seguridad de Cúram es la responsable del almacenamiento en memoria caché de los datos de autorización de un usuario. Debería consultar 5.2, "Memoria caché de seguridad de Cúram", en la página 25 para obtener más detalles.

Las siguientes secciones describen la relación de estos conceptos de autorización y cómo funciona en IBM Cúram Social Program Management.

3.2 Usuarios, roles y grupos

La información de seguridad asociada con una aplicación primero debe organizarse en perfiles de seguridad antes de que se pueda utilizar en un entorno de ejecución. Un perfil de seguridad consiste en un rol de seguridad, uno o varios grupos de seguridad y las asociaciones entre identificadores de seguridad (SID) y elementos protegibles de una aplicación.

A cada usuario autorizado se le asigna un rol de seguridad durante la configuración de seguridad y dichos roles se asocian con numerosos grupos de seguridad. Cada grupo de seguridad está asociado con numerosos identificadores de seguridad. El identificador de seguridad representa los elementos protegibles de IBM Cúram Social Program Management, por ejemplo, un método o un campo. El rol, los grupos y la información de identificador se almacena en la base de datos en numerosas tablas y se configura utilizando la aplicación Data Manager o las pantallas de administración de Cúram.

Esta estructura de datos permite autorizar cada usuario contra cualquier elemento asegurado de una aplicación. Se trata de un método potente y flexible de facilitar autorizaciones a usuarios de Cúram.

Existe un conjunto mínimo de SID (identificadores de seguridad) necesarios para que un usuario pueda utilizar la aplicación Social Program Management Platform. Estos SID (identificadores de seguridad) están asociados con el grupo BASESECURITYGROUP listo para utilizar. Debe consultarse el archivo `EJBServer/components/core/data/initial/handcraftedscripts/Supergroup.sql` para identificar la lista de estos SID (identificadores de seguridad). Este archivo es responsable de enlazar los SID (identificadores de seguridad) con el BASESECURITYGROUP listo para usar.

© Copyright IBM Corp. 2012, 2013

Una forma sencilla de garantizar que todos los usuarios tengan los privilegios de este conjunto de SID (identificadores de seguridad), es crear un único grupo de seguridad para estos y, a continuación, asociar dicho grupo de seguridad con cada rol de seguridad en el sistema.

3.3 Identificadores de seguridad (SID)

A cada elemento asegurado en IBM Cúram Social Program Management se le adjudica un SID (identificador de seguridad) que es exclusivo en toda la aplicación.

El proceso de autorización se crea en la infraestructura y una vez se han identificado los elementos protegibles, el resto es manejado por generadores de códigos, scripts y las pantallas de administración de Cúram. El análisis de los elementos que deben protegerse es un proceso manual que debe ser realizado por un desarrollador o administrador de seguridad. Esta sección explica la infraestructura disponible para configurar la autorización.

El primer tipo de autorización que debe tenerse en cuenta es el método de proceso (fachada) también conocido como *seguridad a nivel de función* . En el modelo de Cúram, un desarrollador puede decidir si la seguridad está activa o no durante en el nivel de método de proceso. Esta opción sólo se aplica a objetos de proceso de negocio (BPO) pues que encapsulan las llamadas expuestas al cliente. Los métodos de objeto de entidad no están incluidos en el proceso de autorización.

Existen numerosos tipos de SID (identificadores de seguridad) y éstos incluyen:

- Identificadores de función (FID)
- Identificadores de seguridad de nivel de campo
- Tipos de SID (identificador de seguridad) definidos por el usuario.

3.4 Identificadores de función (FID)

Los identificadores de función (FID) son un tipo especializado de identificador de seguridad (SID) en el que el tipo se establece en FUNCTION. Cuando un método se hace públicamente accesible (estableciendo el estereotipo como fachada en el modelo), se genera un identificador de función para dicho método y la seguridad se activa automáticamente.

Se puede desactivar la seguridad de un método de proceso durante el tiempo de diseño. Debe consultar 11.11, “Cómo desactivar la seguridad de un método de proceso”, en la página 44 para obtener más detalles sobre este tema.

3.5 Identificadores de seguridad de nivel de campo

El SID (identificador de seguridad) de nivel de campo permite que se aplique la autorización a campos específicos en un método accesible públicamente. Durante el tiempo de ejecución, si un usuario no dispone de derechos de acceso para ver el campo que debe visualizarse, se visualiza el contenido del campo como un número de asteriscos (***). Para obtener más información sobre los SID (identificadores de seguridad) de nivel de campo, debe consultar la publicación *Cúram Modeling Reference Guide* (Manual de consulta de modelado de Cúram).

3.6 SID (identificadores de seguridad) definidos por el usuario

En las secciones anteriores hemos descrito

Identificadores de funciones;
Un SID (identificador de seguridad) generado automáticamente de la función del tipo.

SID (identificador de seguridad) de nivel de campo;
Seguridad aplicada a campos específicos en un método.

También existe el concepto de un SID (identificador de seguridad) definido por el usuario. El proceso de autorización es lo suficientemente flexible como para acomodar cualquier elemento protegible de una aplicación de IBM Cúram Social Program Management. El desarrollador puede personalizar efectivamente el proceso de autorizacón definiendo nuevos *tipos* de SID (identificadores de seguridad). Los nuevos tipos representan un elemento conceptual que requiere seguridad. El siguiente método de interfaz de servidor permite que la autorización se invoque directamente en estos nuevos tipos de SID (identificador de seguridad) definidos por el usuario.

```
curam.util.security.Authorisation.isSIDAuthorised()
```

Instantáneo, los SID (identificadores de seguridad) LOCATION y PRODUCT son SID (identificadores de seguridad) de este tipo. La utilización del método anterior permite definir un número ilimitado de tipos de SID (identificador de seguridad). Debería consultar 11.14, “Autorización de Nuevos tipos de SID (identificador de seguridad)”, en la página 45 para obtener más detalles.

3.7 Autorización de tiempo de ejecución

La infraestructura de IBM Cúram Social Program Management realiza comprobaciones de autorización para el cliente web y para el lado del servidor.

3.8 Comprobaciones de autorización de cliente

Antes de que un usuario pueda acceder a un método o campo, el cliente web realiza comprobaciones de autorización antes de que se cargue inicialmente la página. Si el usuario no tiene acceso, la comprobación de autorización del cliente falla y el servidor no se invoca. Esta comprobación se puede configurar en `curam-config.xml` estableciendo la propiedad SECURITY_CHECK_ON_PAGE_LOAD. La sección 3.12.13 Configuración general de la publicación *Cúram Web Client Reference Manual* (Manual de consulta del cliente web de Cúram) para obtener más detalles sobre este tema.

De forma predeterminada, no se registran los errores de autorización del cliente web. Este comportamiento se puede configurar. Debería consultar 11.13, “Control del registro de errores de autorización del cliente”, en la página 45 para obtener más detalles.

3.9 Comprobaciones de autorización de servidor

Para proveer otro acceso a IBM Cúram Social Program Management y donde la comprobación de autorización de cliente web está inhabilitada, el servidor realiza una comprobación de autorización de segundo nivel. Esta comprobación por la parte del servidor siempre registra errores de autorización y la propiedad de cliente no afecta a dicho registro.

El registro de todos los errores de autorización se almacena en la base de datos para permitir que se puedan auditar estos errores en una etapa posterior. La tabla AuthorisationLog contiene el nombre de usuario y el identificador de seguridad de la autorización fallida, además de una indicación de fecha y hora que indica cuando se ha producido el error. Debe consultar 11.15, “Análisis de la tabla de base de datos AuthorisationLog”, en la página 46 para obtener información detallada sobre la tabla AuthorisationLog.

Capítulo 4. Criptografía en Cúram

4.1 Descripción general

En IBM Cúram Social Program Management, el término criptografía hace referencia básicamente a dos tipos de funcionalidad relacionados con la protección y seguridad de los sistemas Cúram:

1. cifrados: para encriptación bidireccional de contraseñas utilizadas en distintos puntos del procesamiento
2. resúmenes: creación unidireccional de hashes (o resúmenes) de contraseñas, p. ej. utilizadas en un inicio de sesión

El usuario puede seleccionar los valores que configuran el comportamiento criptográfico a través de de un archivo de propiedades (`CryptoConfig.properties`) que proporciona el máximo grado de control y seguridad en una instalación de Cúram. Esta flexibilidad proporciona la posibilidad de amoldarse a los diversos estándares de seguridad a medida que estos evolucionan. Consulte Capítulo 12, “Personalización de la criptografía”, en la página 47 para obtener detalles sobre cómo configurar y personalizar la criptografía.

Para los usuarios ya existentes de IBM Cúram Social Program Management se recomienda actualizar las contraseñas del sistema (nuevo cifrado) y de usuario (nuevo resumen) proporcionadas de forma predeterminada en el producto (OOTB) a fin de mejorar la seguridad. Puesto que la actualización de contraseñas es un cambio sustancial, pueden realizarse ambos cambios de forma independiente tal y como se describe en los correspondientes temas. Podrán dejarse tal y como están las contraseñas de sistema y usuario ya existentes, siempre y cuando uno esté dispuesto a asumir un nivel de seguridad inferior por su cuenta y riesgo, lo que no se recomienda.

Las configuraciones criptográficas soportadas son:

1. AES: 128, 192, 256 (que cumplen con FIPS 140-2 y SP800-131a);
2. DES triple de dos claves - DESede: 112 (que cumple con FIPS 140-2);
3. DES triple de tres claves - DESede: 168 (que cumple con FIPS 140-2 y SP800-131a);
4. ninguna configuración criptográfica, lo que se configura eliminando el archivo `CryptoConfig.properties`, en cuyo caso Cúram volverá a los valores criptográficos OOTB anteriores.

En el entorno en que ejecuta Cúram, el servidor de aplicaciones, la base de datos y otros componentes software (p.ej. servidor web, LDAP, etc.) tendrán su propio soporte criptográfico y deberá consultarse la documentación del correspondiente proveedor.

4.2 Cifrado

El cifrado se refiere al proceso de encriptado de contraseñas. Es decir, es un proceso bidireccional que representa valores descifrables. Existen alrededor de una docena de contraseñas encriptadas en diversos archivos de propiedades de Cúram y su encriptado ayuda a mantenerlas protegidas. Se desencriptan en el momento en que es necesario utilizarlas como, por ejemplo, al conectar con el sistema de base de datos.

4.3 Creación de resúmenes

Por creación de resúmenes se entiende el proceso unidireccional de manejo de contraseñas que no requieren descifrado, y se usa para almacenar contraseñas para una comparación posterior (p.ej. en los inicios de sesión de los usuarios en Cúram). Es decir, es un proceso unidireccional que representa valores no descifrables.

© Copyright IBM Corp. 2012, 2013

4.4 Propiedades de criptografía

El archivo `CryptoConfig.properties` de Cúram contiene los valores de la criptografía de cifrado y de resumen. Por tanto, este archivo y todos los archivos que referencia (p.ej. el almacén de claves y la sal) deben considerarse elementos críticos para la seguridad del sistema y se les deberían aplicar los controles de acceso adecuados (p.ej. permisos de archivo), y modificarlos y aislarlos de forma específica cuando se utilicen en sistemas de producción. Es decir, si los detalles de estos archivos llegaran a ser de dominio público, se eliminaría un nivel de protección (si bien no se convertirían en sí mismos en un riesgo de seguridad) y podría ser necesario un cambio criptográfico disruptivo (consulte 12.2, "Personalización del cifrado", en la página 47 y 12.5, "Personalización de un resumen", en la página 49).

Temas relacionados:

- 4.5, "Valores de cifrado de Cúram"
- 4.6, "Valores de resumen de Cúram", en la página 23

4.5 Valores de cifrado de Cúram

De forma predeterminada en el producto (OOTB), se almacenan en formato encriptado diversas contraseñas en archivos de propiedades de Cúram.

La configuración criptográfica de Cúram que va incluida de forma predeterminada en el producto funciona, pero se recomienda modificar dichos valores conforme a los requisitos de seguridad locales. Por ejemplo, puede que los valores OOTB sean válidos para desarrollo, pero se recomienda encarecidamente que se modifiquen en los entornos de producción (p.ej. cambiando la clave secreta de cifrado).

Los valores de configuración del cifrado se almacenan en el archivo `CryptoConfig.properties`. Las propiedades y sus valores son los siguientes:

- `curam.security.crypto.cipher.algorithm`
  - **Valores válidos:** en la documentación de JCE como, por ejemplo: http://docs.oracle.com/javase/6/docs/technotes/guides/security/StandardNames.html#Cipher.Los cifrados soportados son AES y las diversas variantes de triple DES.
  - **Valor predeterminado:** AES (que cumple con FIPS 140-2 y SP800-131a)
- `curam.security.crypto.superseded.cipher.algorithm`
  - **Valores válidos:** consulte `curam.security.crypto.cipher.algorithm`
  - **Valor predeterminado:** ninguno.
  - **Finalidad:** se proporciona para dar flexibilidad al proceso de actualización/migración mediante código personalizado (p.ej. un programa por lotes) a través del API `curam.util.security.EncryptionUtil.decryptSupersededPassword()`.
- `curam.security.crypto.cipher.keystore.location`
  - **Valores válidos:** ruta del archivo del almacén de claves que contiene la clave secreta. Puede ser una ruta absoluta o relativa al classpath (p.ej. `CuramSample.keystore`).
  - **Valor predeterminado:** ninguno
- `curam.security.crypto.cipher.keystore.storepass`
  - **Valores válidos:** los que admite el comando **`keytool`** del JDK.
  - **Valor predeterminado:** password
  - **Finalidad:** especifica la contraseña empleada para acceder al almacén de claves.
- `curam.security.crypto.cipher.provider.class`
  - **Valores válidos:** nombre completo de una clase proveedora de criptografía JCE.
  - **Valor predeterminado:** vacío
  - **Finalidad:** forma opcional de habilitar el uso de un proveedor alternativo que cumpla con los estándares.

Esta funcionalidad de cifrado se aplica a las propiedades tal y como se describe en 4.7, “Contraseñas cifradas”, en la página 24.

Estos valores criptográficos de Cúram están habilitados de forma predeterminada en OOTB y representan los valores que deben afrontar instalaciones de Cúram existentes tal y como se documenta en la *Guía de actualización de Cúram*.

4.6 Valores de resumen de Cúram

Cuando no se invocan solo con identidad, los usuarios de Cúram (tanto internos como externos) se autentican mediante un inicio de sesión basado en formulario y la contraseña especificada en el formulario se resume y se compara con el valor de resumen almacenado en la base de datos para ese usuario.

Nota: Este procesamiento no se aplica a los usuarios autenticados en sistemas de terceros como LDAP.

La configuración criptográfica de Cúram que va incluida de forma predeterminada en el producto funciona, pero se recomienda modificar dichos valores conforme a los requisitos de seguridad locales. Por ejemplo, puede que los valores OOTB sean válidos para desarrollo, pero se recomienda encarecidamente que se modifiquen en los entornos de producción (p.ej. resumir un valor encriptado de sal).

Los valores de configuración del resumen se almacenan en el archivo `CryptoConfig.properties`. Las propiedades y sus valores son los siguientes:

- `curam.security.crypto.digest.algorithm`
  - **Valores válidos:** en la documentación de JCE como, por ejemplo, http://docs.oracle.com/javase/6/docs/technotes/guides/security/StandardNames.html#MessageDigest. Los resúmenes soportados son las variantes SHA (1, 256, etc.) y MD5.
  - **Valor predeterminado:** SHA-256 (que cumple con FIPS 140-2 y SP800-131a)
  - **Finalidad:** especificar el algoritmo de resumen.
- `curam.security.crypto.digest.salt.location`
  - **Valores válidos:** una ruta que identifique el archivo que contiene la sal de resumen secreto encriptada.
  - **Valor predeterminado:** ninguno.
  - **Finalidad:** un archivo opcional que especifica la sal (encriptada) utilizada para crear los resúmenes.
- `curam.security.crypto.digest.iterations`
  - **Valores válidos:** 0 o un entero positivo.
  - **Valor predeterminado:** 0
  - **Finalidad:** por lo general, cuanto más alto es el valor, mayor es la seguridad, pero a costa de penalizar el rendimiento (p.ej. en tiempo de inicio de sesión).

Hay un conjunto de correspondientes propiedades "reemplazadas" que flexibilizan la migración de un conjunto de valores o estándares de resumen a otro. Las siguientes propiedades se corresponden con las anteriores y tienen una función similar a ellas, pero las utiliza la funcionalidad de encriptado de Cúram para soportar los valores nuevos y antiguos en el momento de migrar.

- `curam.security.crypto.superseded.digest.algorithm`
- `curam.security.crypto.superseded.digest.salt.location`
- `curam.security.crypto.superseded.digest.iterations`

La propiedad `curam.security.convertsupersededpassworddigests.enabled` controla la utilización y el comportamiento de las propiedades reemplazadas como se gestionan en la interfaz de usuario de la

administración de propiedades. Consulte 12.7, "Cómo utilizar los valores de resumen sustituidos durante un período de migración", en la página 50 para obtener información adicional relativa a la utilización de propiedades reemplazadas.

4.7 Contraseñas cifradas

Las siguientes contraseñas están encriptadas mediante cifrado en Cúram:

- `Bootstrap.properties`:
  - `curam.db.password` - contraseña de la base de datos
  - `curam.searchserver.sync.password` - consulte *Servidor de búsquedas genérico de Cúram* para obtener información adicional
- `AppServer.properties`: (este archivo de propiedades se utiliza normalmente para configurar servidores de pruebas y no es adecuado para entornos de producción).
  - `security.password` - contraseña de la consola de administración del servidor de aplicaciones
  - `curam.security.credentials.async.password` - sustituye a `runas.password property`
- `Application.prx` - las descripciones de propiedades individuales están documentadas en las propiedades de la interfaz de usuario de la administración de propiedades de Curam:
  - `curam.security.credentials.dbtojms.password` - (junto con curam.security.credentials.dbtojms.username), que sustituye las API de interfaz curam.omega3.DBtoJMSCredentialsIntf usadas anteriormente para proporcionar credenciales personalizadas BD a JMS
  - `curam.security.credentials.ws.password` (junto con `curam.security.credentials.ws.username`), que sustituye los valores de credenciales predeterminadas de servicios web predeterminados en tiempo de compilación.
  - `curam.meeting.request.reply.password` - (una contraseña SMTP)
  - `curam.ldap.password`
  - `curam.citizenworkspace.password.protection.key`
- `BIBootstrap.properties` - solo para los usuarios de BIRT; consulte la *Guía del desarrollador BIRT de Cúram Business Intelligence*:
  - `curamsource.db.password`
  - `central.db.password`
  - `centraldm.db.password`
- Servicios web - consulte :
  - `ws_inbound.xml` - `<ws_service_password>`
  - `services.xml` - `<parameter name="jndiPassword">`
- CTM - Cúram Transport Manager:
- La columna Password de la tabla TargetSystemService contiene una contraseña encriptada

Capítulo 5. Almacenamiento en memoria caché de datos de seguridad

5.1 Visión general

Este capítulo describe la memoria caché de seguridad de Cúram, que almacena todos los datos de autorización de un usuario. En este capítulo también se incluyen detalles sobre la memoria caché de WebSphere y sobre cómo afecta a la autenticación de un usuario durante el inicio de la sesión.

5.2 Memoria caché de seguridad de Cúram

La infraestructura almacena en memoria caché la información de seguridad de las tablas de base de datos que soportan los perfiles mencionados en 3.2, "Usuarios, roles y grupos", en la página 17. Se realiza para optimizar la búsqueda y la recuperación de datos durante el proceso de autorización.

Para optimizar el rendimiento, la memoria caché se carga a demanda a medida que van llegando solicitudes de autorización de seguridad en la aplicación y es un recurso compartido. Para el código de aplicación, la memoria caché es un recurso protegido y no se puede acceder directamente. Está accesible, sólo para consulta, mediante la interfaz de autorización (`curam.util.security.Authorisation`) que permite que un desarrollador implemente un procedimiento de autorización personalizado. Debe consultar 11.14, "Autorización de Nuevos tipos de SID (identificador de seguridad)", en la página 45 para obtener más detalles sobre este tema.

Cuando la propiedad curam.security.casesensitive está establecida en false, la memoria caché de seguridad almacenará todos los nombres de usuario en mayúsculas y todas las consultas a la memoria caché cambiarán automáticamente el nombre de usuario especificado en su equivalente en mayúsculas. Vale la pena tener en cuenta que la existencia de nombres de usuario no sensibles a mayúsculas y minúsculas duplicados desencadenará errores muy graves durante la inicialización de la memoria caché de seguridad. Debe consultar 10.4, "Cómo cambiar la sensibilidad a mayúsculas y minúsculas del nombre de usuario", en la página 39 para obtener más información acerca de este tema.

5.3 Renovación de la memoria caché

Puesto que los datos de seguridad son tan importantes para el funcionamiento de IBM Cúram Social Program Management, la memoria caché debe renovarse cada vez que se realicen cambios en las tablas de base de datos relacionadas con la seguridad. La renovación de la memoria caché de seguridad de Cúram es un proceso asíncrono.

5.4 Error de renovación de memoria caché

La renovación de la memoria caché de la seguridad de Cúram es desencadenada por un rearranque de la aplicación o por el administrador del sistema (sysadmin) mediante las pantallas de administración de Cúram, por lo que el administrador no recibe ningún feedback si falla la recarga de la memoria caché. Puede resultar incómodo tener que comprobar los registros del sistema o verificar manualmente la aplicación después de una renovación para verificar su posible éxito. Por ello, es recomendable que se implemente la interfaz de devolución de llamada opcional para que facilite un feedback si se produce un error de recarga de la memoria caché. Debería consultar 10.8, "Cómo añadir la interfaz de devolución de llamada de error de renovación de memoria caché", en la página 40 para obtener más detalles.

© Copyright IBM Corp. 2012, 2013

5.5 Comportamiento del almacenamiento en memoria caché de WebSphere

WebSphere almacena la información del usuario y las credenciales en su propia memoria caché de seguridad. El módulo de inicio de sesión de Cúram no se invocará hasta que una entrada de usuario sea válida en esta memoria caché. El tiempo de invalidación predeterminado de esta memoria caché de seguridad es de diez minutos, durante los que el usuario va a estar inactivo.

Por ejemplo, la primera vez que un usuario inicie la sesión en la aplicación desde el cliente web, se le va a solicitar su nombre de usuario y contraseña. El módulo de inicio de sesión de Cúram se invocará y autenticará la información especificada. Si el mismo usuario abre un segundo navegador web nuevo e intenta acceder a la aplicación, el sistema le volverá a solicitar su nombre de usuario y contraseña. Cuando WebSphere recibe esta información, consulta la memoria caché de seguridad para determinar si el nombre de usuario y la contraseña ya existen en dicha memoria caché. Si ya existen y la contraseña coincide, WebSphere no consultará los módulos de inicio de sesión.

El impacto de este comportamiento es que toda modificación en las restricciones de cuenta de usuario o contraseña no tendrá efecto hasta que el usuario se haya invalidado en la memoria caché de seguridad de WebSphere.

Puede obtener información adicional consultando el correspondiente *Information Center de WebSphere Application Server*.

Capítulo 6. Seguridad para clientes alternativos

6.1 Visión general

Hay procesos que puede que no estén asociados con un usuario específico que haya iniciado la sesión. Estos incluyen clientes alternativos, por ejemplo, procesos no de web como el procesamiento por lotes, servicios web y procesamientos aplazados. Como debe autenticarse cualquier proceso que interactúe con una aplicación de IBM Cúram Social Program Management, debe existir un usuario válido para cada uno de estos procesos. Las siguientes secciones proporcionan detalles sobre los usuarios que deben existir en la tabla Usuarios de Cúram, además de información sobre los procesos que dependen de estos usuarios.

6.2 Usuarios de Cúram obligatorios

Siempre debe existir un número de usuarios determinado en la tabla de base de datos de usuarios de Cúram. Estos usuarios resultan necesarios para los procesos de aplicaciones tales como flujos de trabajo y procesos aplazados. Si no existen estos usuarios, a continuación la autenticación falla y seguidamente fallan estos procesos.

Los nombres de usuario y las contraseñas de cada uno los abajo procesados son las credenciales instantáneas predeterminadas y es recomendable que se cambien por motivos de seguridad.

Estos usuarios incluyen:

- SYSTEM

  El usuario SYSTEM es el usuario bajo el que se ejecutan mensajes JMS. Este usuario debe existir y el nombre de usuario es sensible a mayúsculas y minúsculas. Debería consultar 6.5, “Mensajería de JMS”, en la página 28 para obtener más detalles.
- DBTOJMS

  El usuario DBTOJMS es el usuario predeterminado bajo el que se ejecuta el desencadenante de la base de datos de JMS (DBToJMS) para un procesamiento por lotes. Este usuario debe existir y el nombre de usuario es sensible a mayúsculas y minúsculas. Debería consultar 6.4, “Proceso por lotes”, en la página 28 para obtener más detalles.
- WEBSVCS

  El usuario WEBSVCS es el usuario predeterminado bajo el que se ejecutan servicios web. Este usuario debe existir y el nombre de usuario es sensible a mayúsculas y minúsculas. Debería consultar 6.3, “Servicios web” para obtener más detalles.

6.3 Servicios web

En el caso de Apache Axis2 (la implementación recomendada para servicios web) existen credenciales predeterminadas para la autenticación. Un usuario tiene la capacidad de cambiar estas credenciales a nivel global o por servicio, si así resulta necesario. Para garantizar que los servicios web no sean vulnerables ante una infracción en la seguridad, este usuario predeterminado no dispone de autorización para acceder a servicios web de forma predeterminada. Para obtener la autorización, debe asociarse un servicio web con un grupo de seguridad y, a su vez, con un rol de seguridad que está enlazado con el usuario (por ejemplo, WEBSVCS) a fin de permitir el acceso. El proceso de asegurarse de que el usuario esté autorizado es manual. Consulte la sección *Personalización del tiempo de ejecución del receptor* en la publicación *Cúram Web Services Guide* (Guía de servicios web de Cúram) para obtener más detalles sobre servicios web y también el capítulo sobre autorizaciones de este manual.

En el caso de Apache Axis 1.4, es decir, servicios web heredados, una vez se ha modelado un proceso como un servicio web, dicho servicio web se registra automáticamente en la aplicación utilizando las

© Copyright IBM Corp. 2012, 2013

credenciales predeterminadas. Este usuario predeterminado se configura automáticamente para la autorización, es decir, el usuario tendrá acceso al servicio web creado. Por ello, recomendamos tener precaución al establecer una clase visible como un servicio web. Consulte la sección *Servicios web de entrada heredados* en la publicación *Cúram Web Services Guide* (Guía de servicios web de Cúram).

Existen numerosos temas relacionados con la seguridad de servicios web, por ejemplo, cómo cifrar datos y la utilización de Rampart. Debe consultar la publicación *Cúram Web Services Guide* (Guía de servicios web de Cúram) para obtener más información sobre estos temas.

6.4 Proceso por lotes

Puesto que el Lanzador de lotes no requiere que el servidor de aplicaciones esté en ejecución, no realiza ninguna autenticación ni autorización a nivel de aplicación. Sólo debe realizar la autenticación contra la base de datos. Las mismas credenciales que utiliza el servidor de aplicaciones (ubicadas en `%SERVER_DIR%/project/properties/Bootstrap.properties`) son las que utiliza el Lanzador de lotes para conectarse con la base de datos y ejecutar programas por lotes.

El Lanzador de lotes o programas por lotes puede desencadenar opcionalmente el servidor de aplicaciones para iniciar una transferencia DB a JMS. Ello implica iniciar la sesión e invocar un método en el servidor, que a su vez requiere un nombre de usuario y contraseña válidos. De forma predeterminada, la operación de transferencia DB a JMS utiliza credenciales predeterminadas; por ello, la cuenta DBTOJMS debe existir en la tabla Usuarios de Cúram y debe estar habilitada y asignada al rol "SYSTEMROLE" para permitir la autorización. La transferencia de DB a JMS de entorno local es el entorno local predeterminado de este usuario tal como se especifica en el campo "defaultLocale" de la tabla Usuarios.

Debe consultar la sección Consideraciones de seguridad de la guía *Cúram Batch Processing Guide* (Guía del proceso por lotes de Cúram) para obtener más detalles sobre cómo cambiar el usuario para la transferencia DB a JMS.

La propiedad property batch.username se puede utilizar para especificar el nombre de usuario de las operaciones ejecutadas por el Lanzador de lotes. Se establece utilizando el parámetro -D. Por ejemplo: build runbatch -Dbatch.username=admin

6.5 Mensajería de JMS

Los mensajes de JMS se utilizan para poder comunicarse por procesos aplazados y del flujo de trabajo. Puesto que los mensajes de JMS son desencadenados por el servidor de aplicaciones y deben interactuar con la aplicación de IBM Cúram Social Program Management, deben existir credenciales válidas de Cúram. La cuenta de usuario SYSTEM debe existir en la tabla Usuarios de Cúram y debe estar habilitada y asignada al rol "SYSTEMROLE" para garantizar la autorización. El entorno local de mensajes de JMS es el entorno local predeterminado para este usuario, tal como se especifica en el campo 'defaultLocale' en la tabla Usuarios.

Se puede cambiar el nombre de usuario SYSTEM durante o después del despliegue de la aplicación. Para obtener más información consulte la publicación *Cúram Server Deployment Guide* (Guía de despliegue del servidor de Cúram) del servidor de aplicaciones relevante.

6.6 Procesamiento aplazado

Un proceso aplazado en IBM Cúram Social Program Management es un método de negocio que se invoca asíncronramente. Un proceso aplazado interactúa con la aplicación, por lo que deben existir credenciales de Cúram válidas. La cuenta de usuario SYSTEM debe existir en la tabla de usuarios de Cúram y debe estar habilitada y asignada al rol "SYSTEMROLE" para garantizar la autorización. El entorno local para procesos aplazados es el entorno local predeterminado de dicho usuario tal como se especifica en el campo "defaultLocale" en la tabla Usuarios. En el caso de prueba de unidad fuera de línea de procesos

aplazados, el nombre de usuario aparece en blanco y el entorno local efectivo es el entorno local predeterminado del servidor de IBM Cúram Social Program Management.

Capítulo 7. Aplicaciones de usuario externo

7.1 Visión general

En la configuración predeterminada de IBM Cúram Social Program Management, la aplicación instantánea está habilitada para usuarios internos. Los usuarios internos son usuarios que existen en la tabla de base de datos de usuarios de Cúram. Un usuario interno típico sería el caso de un trabajador que crea y gestiona reclamaciones para participantes y tiene acceso completo a la aplicación. La infraestructura proporciona funcionalidad para autenticar y autorizar a estos usuarios internos.

También existe el concepto de un usuario que necesita acceder de forma segura a partes de la aplicación de IBM Cúram Social Program Management. Estos usuarios no forman parte de la organización. Su acceso está limitado. Estos usuarios se consideran usuarios externos y la autenticación de dichos usuarios se puede personalizar completamente utilizando el punto de gancho de seguridad de acceso externo proporcionado. Puesto que los usuarios externos se procesan de forma distinta a los usuarios internos, se requiere una aplicación web específica para los usuarios externos.

7.2 Aplicaciones de usuario externo

Al desarrollar una aplicación para un usuario externo, debe implementar lo siguiente:

- Una aplicación cliente de usuario externo, es decir, un archivo EAR independiente que contenga la aplicación de cliente web.
- Un `logon.jsp` personalizado, en el que la aplicación externa debe pasar un parámetro user_type indicando que un usuario externo está iniciando la sesión.
- Debe proporcionarse una clase personalizada que extienda `curam.util.security.PublicAccessUser`, lo que requiere la implementación de la interfaz `curam.util.security.ExternalAccessSecurity`. Esta clase abstracta contiene métodos responsables de la autenticación y la autorización de un usuario externo.

Así como la existencia de tipos de usuarios internos y externos. También puede haber distintos tipos de usuarios externos. Por ejemplo, puede haber un usuario externo del tipo "PUBLIC" que haya limitado el acceso a una aplicación externa. Puede haber otro usuario externo del tipo "PROVIDER" que sea un usuario externo registrado. La posibilidad de tener distintos tipos de usuarios externo proporciona más flexibilidad dentro de una aplicación externa y permite un control más preciso sobre la autenticación del usuario externo basándose en el tipo de usuario externo.

7.3 Ámbito del usuario

Hay dos tipos (o ámbitos) distintos de usuarios dentro de la aplicación de IBM Cúram Social Program Management: internos y externos. El tipo de usuario se determina de una de las siguientes maneras:

- mediante la memoria caché de seguridad de Cúram;

  si existe el usuario en la memoria caché de seguridad de Cúram, se asume que el tipo es interno. Si el usuario no existe en la memoria caché, se asume que el tipo es externo. En este caso (que es el comportamiento predeterminado), todos los nombres de usuario, tanto los internos y externos, deben ser exclusivos.
- mediante la interfaz personalizada `UserScope`;

  si se ha implementado la interfaz personalizada `UserScope`. Esta interfaz personalizada tiene prioridad frente a la comprobación de un usuario en la memoria caché de seguridad de Cúram para determinar el tipo de usuario. Consulte 13.15, “Determinación de si un usuario es interno o externo utilizando la interfaz UserScope”, en la página 60 para obtener más detalles.

© Copyright IBM Corp. 2012, 2013

Cuando el tipo de usuario es externo, se utilizará la implementación del métodocuram.util.security.ExternalAccessSecurity.getSecurityRole() para determinar el rol de usuario en lugar de los roles de seguridad internos. Debe consultar 13.8, “Autorización de un usuario externo”, en la página 57 para obtener más detalles sobre este método.

Para dar soporte a métodos alternativos para determinar si un usuario es interno o externo, dispone de la interfaz personalizada UserScope. Consulte 13.15, “Determinación de si un usuario es interno o externo utilizando la interfaz UserScope”, en la página 60 para obtener más detalles.

7.4 Despliegue de una aplicación externa

Al desplegar una aplicación en un servidor de aplicaciones, la configuración de seguridad del servidor de aplicaciones es aplicable a todas las aplicaciones de IBM Cúram Social Program Management desplegadas en la instancia del servidor de aplicaciones. Por ello, debe tenerse precaución al considerar la arquitectura del despliegue para más de una aplicación. Es importante al decidir si una aplicación interna y externa se va a desplegar en la misma instancia del servidor de aplicaciones.

Un ejemplo de algunas de las consideraciones que hay que tener en cuenta son:

- ¿se está utilizando "sólo de identidad" para usuarios internos?
- ¿se utiliza un mecanismo de autenticación alternativo, por ejemplo LDAP?;
- ¿LDAP va a autenticar tanto los usuarios internos como los externos?

Las respuestas a las preguntas de arriba afectarán a la configuración de las propiedades del servidor de aplicaciones (es decir, a las propiedades especificadas en el archivo AppServer.properties) que afectan al comportamiento del módulo de inicio de sesión JAAS de Cúram. Estas consideraciones también determinarán la implementación de la clase curam.util.security.PublicAccessUser y la interfaz curam.util.security.ExternalAccessSecurity para usuarios externos.

Las propiedades del servidor de aplicaciones en el módulo de inicio de sesión JAAS de Cúram permiten un control preciso de la autenticación de tipos de usuario. Los usuarios externos y los usuarios externos se pueden autenticar de forma diferente, así como se hace para los distintos tipos de usuarios externos en una situación en la que las aplicaciones internas y externas se despliegan en el mismo servidor de aplicaciones. Estas propiedades incluyen lo siguiente:

- curam.security.user.registry.disabled.types ;

  Establecer esta propiedad en una lista separada por comas de tipos de usuario para los que *no* se consultará el registro de usuarios del servidor de aplicaciones, es decir, la implementación dentro del método PublicAccessUser.authenticate() es la responsable de autenticar al usuario externo de este tipo. Por ejemplo, se puede configurar el LDAP para que sea el registro de usuarios.
- curam.security.user.registry.enabled.types.

  Establecer esta propiedad en una lista separada por comas de tipos de usuario para los que se *va* a consultar el registro de usuarios, es decir, la implementación dentro del método PublicAccessUser.authenticate() no autentica completamente al usuario. El registro de usuarios será el responsable de autenticar este tipo de usuario externo. Por ejemplo, se puede configurar el LDAP como el registro de usuarios, en cuyo caso, el LDAP podría ser el responsable de la autenticación de estos tipos de usuario externo.

Estas propiedades dependen de la implementación de la clase curam.util.security.PublicAccessUser y de la interfaz ExternalAccessSecurity.

Tenga en cuenta los siguientes requisitos de proyecto a modo de ejemplo:

- Un usuario interno debe autenticarse con LDAP.
- Un usuario externo del tipo 'EXT_PUBLIC' debe autenticarse con IBM Cúram Social Program Management y no con LDAP;

- Un usuario externo del tipo 'EXTERNAL' debe autenticarse sólo con LDAP y no con IBM Cúram Social Program Management.
- Las aplicaciones internas y externas se despliegan en la misma instancia del servidor de aplicaciones.

Los siguientes valores podrían aplicarse al ejemplo anterior:

- curam.security.check.identity.only establecido en `true`;
- curam.security.user.registry.disabled.types=EXT_PUBLIC.

Además de establecerse las propiedades, la extensión `PublicAccessUser` (y la implementación de `curam.util.security.ExternalAccessSecurity`) deberá tener la lógica de obtención de los distintos tipos de usuarios externos y el modo en que se autentican.

Capítulo 8. Utilización de un inicio de sesión único

8.1 Visión general

El número de aplicaciones en una empresa a menudo da como resultado un aumento del número de nombres de usuario y contraseñas que se están utilizando, lo que redunda en una pobre experiencia por parte del usuario y en costes adicionales para mantenerlos. Varios nombres de usuario y contraseñas también comprometen la seguridad puesto que como usuarios eligen contraseñas demasiado simples o escriben sus contraseñas en sitios fáciles de encontrar. Para administradores del sistema, las aplicaciones adicionales redundan en un aumento en el esfuerzo del mantenimiento del directorio y en más llamadas al centro de atención al cliente para solicitar un restablecimiento de las contraseñas. Algunos de los problemas causados por aplicaciones adicionales se pueden resolver utilizando la funcionalidad de inicio de sesión único. La funcionalidad de inicio de sesión único (SSO) permite acceder a múltiples aplicaciones seguras realizando una sola autenticación.

Nota: El término "seguro" hace referencia a aplicaciones que requieren que los usuarios se autentiquen antes de acceder a la funcionalidad

El inicio de sesión único se soporta en servidor de aplicaciones, permitiendo que se utilicen mecanismos alternativos en el módulo de inicio de sesión de Cúram. La implementación de una solución de inicio de sesión único (SSO) es la responsabilidad de la implementación personalizada. Se recomienda utilizar una herramienta de terceros como, por ejemplo, IBM Tivoli ™ o CA SiteMinder.

Este capítulo describe las propiedades del servidor de aplicaciones que permite utilizar una solución de inicio de sesión único.

8.2 Inicio de sesión único en WebSphere

Cuando se requiere un inicio de sesión único con WebSphere, se puede lograr utilizando el mecanismo de Lightweight Third Party Authentication (LTPA) de WebSphere y módulos de inicio de sesión personalizados adicionales. El protocolo de LTPA desencadena la creación de una señal para un usuario autenticado. En WebSphere, se genera una señal una vez se han añadido las credenciales para un usuario autenticado. A continuación, esta señal se utiliza para recuperar información de identidad para un usuario autenticado en un entorno de inicio de sesión único.

La seguridad se implementa como un módulo de inicio de sesión de Cúram dentro de una cadena de módulos de inicio de sesión configurada en WebSphere. Se espera que como mínimo uno de estos módulos de inicio de sesión sea el responsable de añadir las credenciales del usuario. De forma predeterminada, el módulo de inicio de sesión de Cúram añade las credenciales del usuario autenticado. En consecuencia, el registro de usuarios de WebSphere configurado manejado por un módulo de inicio de sesión subsiguiente no añade más credenciales. El método recomendado para implementar una solución de inicio de sesión único es añadir un módulo de inicio de sesión personalizado en algún lugar a lo largo de la cadena de módulos de inicio de sesión.

Existe la posibilidad de inhabilitar la adición de credenciales para un usuario no autenticado, por lo que se permite que se implemente una solución de inicio de sesión único.

El módulo de inicio de sesión JAAS de Cúram para WebSphere comprueba si existe un token LTPA dentro de WebSphere utilizando la devolución de llamada WSCredTokenCallbackImpl para WebSphere. Si este token existe y es válido, entonces el módulo de inicio de sesión de Cúram no realiza ninguna autenticación.

© Copyright IBM Corp. 2012, 2013

Las credenciales se pueden añadir al registro de usuarios de WebSphere. Las credenciales incluyen información de autenticación sobre el inicio de sesión del usuario, incluyendo el identificador exclusivo del usuario. WebSphere comprueba que existan las credenciales para un usuario después de haber ejecutado los módulos de inicio de sesión del sistema configurado. Si existen las dichas credenciales, no se consultará el registro de usuarios de WebSphere. El módulo de inicio de sesión JAAS de Cúram no añade las credenciales si se han establecido los siguientes valores:

- la propiedad curam.security.check.identity.only está establecida en true.
- la propiedad curam.security.user.registry.enabled está establecida en true.

Tal y como se ha mencionado en 7.4, “Despliegue de una aplicación externa”, en la página 32, hay propiedades relacionadas con el tipo de usuario externo que controlan si las credenciales se añaden a WebSphere para un tipo de usuario externo específico. Estas incluyen:

- la propiedad curam.security.user.registry.enabled.types.
- la propiedad curam.security.user.registry.disabled.types.

Estas propiedades proporcionan un control preciso de la autenticación para tipos de usuario externos.

En el caso en el que el módulo de inicio de sesión JAAS de Cúram no añada credenciales, se pedirá al registro de usuarios de WebSphere que intente añadir credenciales para el usuario.

8.3 Inicio de sesión único de WebLogic Server

Cuando se requiere un inicio de sesión único en WebLogic Server, se puede lograr utilizando el proveedor de autenticación de WebLogic Server o un proveedor de autenticación personalizado. Consulte la documentación de WebLogic Server para obtener más información sobre los proveedores de autenticación. WebLogic Server espera credenciales/entidades de seguridad y el grupo al que pertenece el usuario para que el proveedor de autenticación configurado los añada. Para una solución de inicio de sesión único, el módulo de inicio de sesión JAAS de Cúram no añade credenciales al asunto de JAAS para permitir que un proveedor de autenticación alternativo sea responsable de añadir credenciales.

Las credenciales no se añaden si se han definido los siguientes valores:

- curam.security.check.identity.only está establecido en true.
- curam.security.user.registry.enabled está establecido en true.

Tal como se ha mencionado en 7.4, “Despliegue de una aplicación externa”, en la página 32, hay propiedades relacionadas con el tipo de usuario externo que controlan si las credenciales se añaden a WebLogic Server para un tipo de usuario externo específico. Estos incluyen:

- la propiedad curam.security.user.registry.enabled.types
- la propiedad curam.security.user.registry.disabled.types

Estas propiedades proporcionan un control preciso de la autenticación para tipos de usuario externos.

La responsabilidad de añadir credenciales se deja a otro proveedor de autenticación, es decir, el proveedor de autenticación principal para autenticar al usuario. En un escenario de inicio de sesión único, únicamente uno de los proveedores de autenticación necesita añadir credenciales al asunto de JAAS durante el método commit() del módulo de inicio de sesión para un usuario

Capítulo 9. Otras consideraciones de seguridad

9.1 Visión general

Otro asunto importante de seguridad es proteger el contenido tal como se especifica, visualiza y transfiere por toda la red de la aplicación de IBM Cúram Social Program Management. La configuración predeterminada utiliza la SSL proporcionada por el servidor de aplicaciones para proteger el contenido tal como se transfiere.

Además, durante el ciclo de vida del desarrollo, los principales productos industriales se utilizan para supervisar con regularidad si se producen vulnerabilidades de seguridad en la aplicación. Algunos ejemplos de tales vulnerabilidades potenciales incluyen: scripts de sitios cruzados e inyección de SQL. Tales amenazas se resuelven dentro de la infraestructura cuando se descubren.

Es recomendable que los clientes realicen una supervisión de seguridad parecida en sus aplicaciones.

9.2 Valores SSL de la aplicación

SSL está activado de forma predeterminada para acceder en la aplicación web. Ello garantiza una conexión SSL segura entre el cliente y el servidor y también garantiza que se cifren datos. SSL se activa para el cliente mediante los valores del archivo `web.xml` para la aplicación de cliente web. Los valores de WebLogic Server y WebSphere activan SSL a nivel de servidor de aplicaciones. Estos valores para los servidores de aplicaciones se definen mediante los scripts de configuración de IBM Cúram Social Program Management.

Importante: Los scripts de configuración garantizan que SSL esté activo de forma predeterminada pero, no obstante, se trata de una configuración predeterminada que debe actualizarse y deben establecerse certificados nuevos para el protocolo SSL.

Es recomendable dejar SSL activo para acceder a la aplicación de IBM Cúram Social Program Management pero, en función de configuraciones de proyecto específicas, es posible que sea necesario desactivar SSL para la aplicación.

Se puede desactivar SSL, pero no es recomendable. Debería consultar 10.9, “Desactivación de valores SSL de la aplicación”, en la página 41 para obtener más detalles.

© Copyright IBM Corp. 2012, 2013

Capítulo 10. Personalización de la autenticación

10.1 Personalización de la página de inicio de sesión

La pantalla de inicio de sesión instantáneo predeterminada viene representada por el archivo `logon.jsp` ubicado en el directorio `lib/curam/web/jsp` de Client Development Environment for Java (CDEJ). El archivo `logon.jsp` se puede personalizar creando una copia del archivo listo para usar y colocándolo en una carpeta `webclient/components/<custom>/WebContent`, donde <*personalizado*> representa el nombre del componente de cliente web personalizado.

La sección sobre las páginas de inicio de sesión en la publicación *Cúram Web Client Reference Manual* (Manual de consulta de cliente web de Cúram) contiene directrices sobre lo que debe conservarse en archivo `logon.jsp` y debe consultarse para obtener más detalles.

10.2 Aplicación de un estilo a la página de inicio de sesión

Los cambios en el estilo se pueden aplicar a `logon.jsp` de modo habitual, es decir, añadiendo la hoja de estilo en cascada relevante a cualquier archivo .css en el componente personalizado. Debe consultar la publicación *Cúram Web Client Reference Manual* (Manual de consulta del cliente web de Cúram) para obtener detalles sobre el estilo.

10.3 Habilitación de nombres de usuario con caracteres ampliados para WebLogic Server

Si no se está utilizando el servidor de aplicaciones WebLogic Server, esta sección puede ignorarse.

En el caso de que tenga nombres de usuario y contraseñas de Cúram con caracteres ampliados (por ejemplo, "üßer") WebLogic Server proporciona un atributo de propiedad, **j_character_encoding**, que debe añadirse a la página de inicio de sesión basado en formulario **logon.jsp**. Consulte la documentación de WebLogic Server para obtener más información. El atributo debe añadirse al elemento de tabla en el archivo `logon.jsp`, tal como se muestra abajo: 10.3, “Habilitación de nombres de usuario con caracteres ampliados para WebLogic Server”.

```
<input type="hidden" name="j_character_encoding" value="UTF-8"/>
```

Figura 6. Soporte de WebLogic Server para inicios de sesión con caracteres ampliados

10.4 Cómo cambiar la sensibilidad a mayúsculas y minúsculas del nombre de usuario

La propiedad curam.security.casesensitive controla la sensibilidad a mayúsculas y minúsculas de los nombres de usuario. De forma predeterminada, está establecida en `true` en el archivo `Application.prx`. Si se establece en `false` en al archivo `Application.prx`, hace que los mecanismos de autenticación y autorización ignoren el caso del nombre de usuario.

El capítulo *Valores de configuración de Cúram* de la publicación *Cúram Server Developer's Guide* (Guía del desarrollador del servidor de Cúram) encontrará más detalles en el archivo `Application.prx`.

© Copyright IBM Corp. 2012, 2013

10.5 Cómo añadir verificaciones personalizadas al proceso de autenticación

Para añadir verificaciones personalizadas, debe haberse implementado la interfaz `curam.util.security.CustomAuthenticator`. Esta interfaz contiene un método: `authenticateUser()`. El método `authenticateUser()` se invoca tanto para la autenticación predeterminada como para la autenticación sólo de identidad. Se espera que los resultados de este método sean una entrada de la tabla de códigos `curam.util.codetable.SECURITYSTATUS`. Si se produce una autenticación satisfactoria, el resultado debe ser `curam.util.codetable.SECURITYSTATUS.LOGIN`

En caso de error de autenticación el sistema puede devolver cualquier cosa, incluso nulo. Se recomienda, no obstante, usar otra código de la tabla de códigos `curam.util.codetable.SECURITYSTATUS`. Esta tabla de códigos se puede ampliar para que incluya códigos personalizados tal como se explica en el capítulo sobre Tablas de códigos en la publicación *Cúram Server Developer's Guide* (Guía del desarrollador del servidor de Cúram).

Después de invocar las verificaciones personalizadas, el proceso de autenticación actualiza los campos relevantes en la tabla de base de datos de usuarios. Por ejemplo, si el resultado de las verificaciones personalizadas no es `SECURITYSTATUS.LOGIN`, el número de errores de inicio de sesión aumenta en 1 y si se alcanza el umbral de intrusión, se inhabilita la cuenta. De forma alternativa, si el resultado es `SECURITYSTATUS.LOGIN`, los errores de inicio de sesión se restablecen en 0 y se actualiza el último campo de inicio de sesión satisfactorio.

Nota: Cuando se ha habilitado la autenticación de sólo identidad, los campos de la tabla de base de datos de usuarios no se actualizan, independientemente del resultado de la verificación personalizada.

10.6 Configuración del autenticador personalizado

Para configurar la aplicación para que utilice esta ampliación personalizada, la propiedad curam.custom.authentication.implementation en `Application.prx` debe estar establecida según el nombre completo de la clase que implementa la interfaz `CustomAuthenticator`.

El capítulo *Valores de configuración de Cúram* de la publicación *Cúram Server Developer's Guide* (Guía del desarrollador del servidor de Cúram) encontrará más detalles en el archivo `Application.prx`.

10.7 Configuración de la autenticación sólo de identidad

Para configurar la autenticación sólo de identidad, la propiedad curam.security.check.identity.only debe estar establecida en `true` en el archivo `AppServer.properties` antes de ejecutar el destino **`configure`**. También puede establecer esta propiedad una vez se haya desplegado la aplicación mediante la consola del servidor de aplicaciones. Para obtener más información sobre cómo configurar el servidor de aplicaciones, debe consultar las publicaciones *Cúram Server Deployment Guides* (Guías de despliegue del servidor de Cúram) del servidor de aplicaciones que esté utilizando.

10.8 Cómo añadir la interfaz de devolución de llamada de error de renovación de memoria caché

La nueva clase de devolución de llamada debe implementar la interfaz: `curam.util.security.SecurityCacheFailureCallback` en una clase que tenga un constructor predeterminado público. La implementación de la devolución de llamada se registra estableciendo la propiedad de la aplicación curam.security.cache.failure.callback en el nombre de la clase de implementación. Si no se establece la propiedad, no se realiza ningún intento de invocar al manejador de devolución de llamada.

10.9 Desactivación de valores SSL de la aplicación

De forma predeterminada, la SSL está activa para acceder a la aplicación de IBM Cúram Social Program Management. Ello garantiza una conexión SSL segura entre el cliente y el servidor y también garantiza que se cifren datos. La SSL se puede activar y desactivar para el cliente mediante los valores del archivo `web.xml` de la aplicación del cliente web, y a nivel de servidor de aplicaciones mediante los valores de WebLogic Server y WebSphere. Estos valores para los servidores de aplicaciones se configuran a través de los scripts de configuración. Es recomendable dejar la SSL activa para acceder a la aplicación, no obstante, en función de las configuraciones del proyecto específicas, es posible que deba desactivarse la SSL para la aplicación. Las siguientes secciones detallan cómo hacerlo.

10.10 Modificación del archivo web.xml de la aplicación cliente

Se puede modificar cambiando <transport-guarantee> de `CONFIDENTIAL` a `NONE` en el archivo `web.xml`. Tenga en cuenta, que ello no inhabilita el acceso al cliente web mediante HTTPS, pero habilita acceso adicional mediante HTTP. Para obtener más detalles sobre cómo modificar el archivo `web.xml`, debe consultar la sección en *Personalización del descriptor de la aplicación web* en la publicación *Cúram Web Client Reference Manual* (Manual de consulta de cliente web de Cúram). A continuación encontrará un ejemplo de esta propiedad:

```
<user-data-constraint>
         <transport-guarantee>NONE</transport-guarantee>
        </user-data-constraint>
```

10.11 Modificación de la configuración del servidor de aplicaciones

La modificación de la configuración de WebSphere se puede realizar de una de dos formas. El primer método que se describe a continuación es el recomendado.

- Utilice el puerto no seguro existente y configure el predeterminado para Web Services (método recomendado). Es apto tanto para conexiones SSL y no SSL.
  1. Navegue a Entorno -> Hosts virtuales -> host_cliente->Alias de host
  2. Pulse Nuevo e introduzca * en en nombre de host y 9082 para el número de puerto. A continuación, pulse Aceptar
  3. En la página siguiente, pulse Guardar para almacenar el valor nuevo en la configuración del servidor. Tenga en cuenta que el puerto 9082 corresponde al *CuramWebServicesChain* configurado en la aplicación cliente predeterminada y que este puerto ahora es el puerto que se puede utilizar para acceder a la aplicación utilizando HTTP
- Vuelva a utilizar el puerto SSL actual de 9044 :

  El puerto actual se puede configurar como un puerto no seguro. Los pasos para hacerlo se describen en la publicación *Cúram Deployment Guide for WebSphere Application Server* (Guía de despliegue de Cúram para WebSphere Application Server) - Sección A.2.11 Configuración de servidor - Configurar acceso de puerto. Siga los pasos del 7 al 11 incluido. La única diferencia del paso 11, es que el plantilla de cadena de transporte debe establecer en "WebContainer" (y no en contenedor web seguro).

10.12 Análisis de la tabla de base de datos AuthenticationLog

Todos los intentos de autenticación (tanto éxitos como fracasos) se anotan en la tabla de base de datos AuthenticationLog. A continuación encontrará las filas que resultan de interés en esta tabla:

Tabla 1. Contenido del registro de autenticación

Campo	Significado
timeEntered	La indicación de fecha y hora de la entrada en el registro.
userName	El nombre de usuario asociado con el intento de inicio de sesión.

Tabla 1. Contenido del registro de autenticación (continuación)

Campo	Significado
altLogin	Booleano que indica si el nombre de usuario representa un ID de inicio de sesión alternativo. Cuando esta columna es igual a '1' (verdadero), el valor de la columna userName es un ID de inicio de sesión alternativo tal y como se describe en 2.5, "ID de inicio de sesión alternativos", en la página 5; de lo contrario, la columna userName representa el userName de las tablas Users o ExternalUser.
loginFailures	El número de errores de inicio de sesión de este usuario desde su último inicio de sesión satisfactorio.
lastLogin	La fecha y la hora del último inicio de sesión satisfactorio.
loginStatus	El estado del intento de inicio de sesión. Puede ser uno de los siguientes: • LOGIN: Inicio de sesión satisfactorio. • ACCDISABLE: La cuenta se ha inhabilitado explícitamente. • ACCEXPIRED: Se ha alcanzado la fecha de caducidad de la contraseña. • PWDEXPIRED: Se ha excedido el número de días que se ha otorgado al usuario para cambiar su contraseña. • BADUSER: El usuario no existe. • AUTHONLY: Se utiliza en el caso de la autenticación sólo de identidad e indica que sólo se realizarán verificaciones de autorización. • BADPWD: La contraseña especificada era incorrecta. • BREAKIN: Se ha alcanzado un número especificado de contraseñas incorrectas. La cuenta está inhabilitada. • RESTRICTED: El usuario no dispone de acceso al sistema en este momento. • LOGEXPR: Se ha excedido el número de intentos de inicio de sesión que se había otorgado al usuario para cambiar su contraseña. • AMBIGUOUS: El nombre de usuario especificado es ambiguo puesto que es un duplicado no sensible a mayúsculas y minúsculas de otro nombre de usuario.

Se puede utilizar la API `LogAdmin` para consultar la tabla de base de datos AuthenticationLog. Debe consultar la documentación de Java para esta clase para obtener más detalles.

Capítulo 11. Personalización de la autorización

11.1 Visión general

Este capítulo detalla cómo configurar la autorización de usuarios.

11.2 Creación de correlación de datos de autorización

Los datos de autorización de un usuario se pueden configurar utilizando Data Manager (archivos DMX) o mediante pantallas de administración de Cúram. Debe consultar la publicación *Cúram System Configuration Guide* (Guía de configuración del sistema Cúram) para obtener detalles sobre cómo identificar la forma de agrupar la seguridad desde la perspectiva empresarial.

Para crear un nuevo rol de seguridad para el usuario, deben identificarse los identificadores de seguridad (SID) a los que el usuario debe tener acceso. A continuación, estos SID deben organizarse en grupos de SID. El rol, grupos y SID (identificadores de seguridad), una vez identificados, deben establecerse en las tablas de seguridad que representan.

Los datos de seguridad se consideran esenciales para la configuración de una aplicación de IBM Cúram Social Program Management. Como tal, los ejemplos de más abajo describen cómo añadir datos de seguridad al directorio `data/initial` dentro del componente.

11.3 Creación de un nuevo rol de seguridad

Para crear un nuevo rol de seguridad debe añadirse una entrada nueva en la tabla de base de datos SecurityRole estableciendo el atributo `rolename`.

Para hacerlo, cree/añada el/al archivo `SecurityRole.dmx` en `%SERVER_DIR%/components/<custom>/data/initial`, donde `<custom>` es cualquier directorio nuevo creado bajo componentes que se ajusta a la misma estructura de directorio que `components/core`.

11.4 Creación de un nuevo grupo de seguridad

Para crear un nuevo grupo de seguridad, debe añadirse una entrada nueva en la tabla de base de datos SecurityGroup estableciendo el atributo `groupname`.

Para hacerlo, cree/añada el/al archivo `SecurityGroup.dmx` en `%SERVER_DIR%/components/<custom>/data/initial`, donde `<custom>` es cualquier directorio nuevo creado bajo componentes que se ajusta a la misma estructura de directorio que `components/core`.

11.5 Cómo enlazar el grupo de seguridad con el rol de seguridad

El rol de seguridad debe enlazarse con el grupo de seguridad. Para hacerlo, cree una entrada nueva en la tabla SecurityRoleGroup estableciendo los atributos `rolename` y `groupname`.

Para hacerlo, cree/añada el/al archivo `SecurityRoleGroup.dmx` en `%SERVER_DIR%/components/<custom>/data/initial`, donde `<custom>` es cualquier directorio nuevo creado bajo componentes que se ajusta a la misma estructura de directorio que `components/core`.

11.6 Creación del identificador de seguridad (SID)

Para crear un nuevo SID (identificador de seguridad), debe añadirse una entrada en la tabla SecurityIdentifier estableciendo los atributos `sidname` y `sidtype`.

© Copyright IBM Corp. 2012, 2013

Para hacerlo, cree/añada el/al archivo SecurityIdentifier.dmx en %SERVER_DIR%/components/<custom>/data/initial, donde <custom> es cualquier directorio nuevo creado bajo componentes que se ajusta a la misma estructura de directorio que components/core.

11.7 Cómo enlazar el grupo de seguridad con el SID (identificador de seguridad)

Para enlzar el grupo de seguridad con el SID (identificador de seguridad), debe añadirse una entrada a la tabla SecurityGroupSID estableciendo los atributos groupname y sidname.

Para hacerlo, cree/añada el/al archivo SecurityGroupSID.dmx en %SERVER_DIR%/components/<custom>/data/initial, donde <custom> es cualquier directorio nuevo creado bajo componentes que se ajusta a la misma estructura de directorio que components/core.

11.8 Cómo enlazar el rol de seguridad con el usuario

Para asociar datos de autorización con un usuario, el rol de seguridad debe estar enlazado con el usuario.

Para hacerlo, actualice la entrada del usuario especificado en el archivo Users.dmx ubicado en %SERVER_DIR%/components/<custom>/data/initial, donde <custom> es cualquier nuevo directorio creado bajo componentes q ue se ajusta a la misma estructura de directorio que components/core estableciendo el atributo rolename para que sea el rolename tal como se especifica en la tabla SecurityRole.

11.9 Cómo cargar información de seguridad en la base de datos

Una vez se haya introducido toda la información en distintos archivos DMX, debería utilizar el Data Manager para cargar los datos DMX en la base de datos. El capítulo *Data Manager* de la publicación *Cúram Server Developer's Guide* (Guía del desarrollador del servidor de Cúram) proporciona información más detallada.

11.10 Creación de identificadores de función (FID)

Cuando un método se hace públicamente accesible estableciendo el estereotipo para que sea <<fachada>>, la seguridad se activa automáticamente. Ello significa que se genera automáticamente un SID (identificador de seguridad) para dicho método y el distintivo habilitado para la seguridad del método se establece en true . El SID (identificador de seguridad) y su distintivo fidenabled se almacenan en el archivo <ProjectName>_Fids.xml independiente de la base de datos ubicado en el subdirectorio /build/svr/gen/ddl. Este archivo se utiliza para insertar la información del FID en la base de datos mediante Data Manager.

Un FID sigue la convención de nomenclatura de <classname>.<methodname> y la longitud máxima de un FID es de 100 caracteres. Por ejemplo, para un objeto de proceso de negocio denominado ProductEligibility, con dos métodos llamados insertProduct y testProduct, se crean dos FID: ProductEligibility.insertProduct y ProductEligibility.testProduct.

Si la seguridad de un método de proceso está desactivada durante el tiempo de diseño en el modelo, todavía se genera un SID/FID, pero el distintivo habilitado para la seguridad se establece en false. El establecimiento del distintivo habilitado para la seguridad en false significa que no se va a realizar ninguna comprobación de autorización para este método.

11.11 Cómo desactivar la seguridad de un método de proceso

Si se establece la opción Generate_Security (Generar seguridad) en el método de proceso en false en el modelo, se desactiva la seguridad de un método de proceso.

Si la seguridad de un método de proceso está desactivada durante el tiempo de diseño en el modelo, todavía se genera un FID, pero el distintivo habilitado para la seguridad se establece en `false`. El establecimiento del distintivo habilitado para la seguridad en false significa que no se va a realizar ninguna comprobación de autorización para este método.

11.12 Consideraciones de seguridad durante el desarrollo

Es importante tener en cuenta el efecto de estas opciones de diseño al implementar la seguridad durante el desarrollo de una aplicación de IBM Cúram Social Program Management. Se trata de la primera y la última línea de defensa contra el acceso no autorizado a la funcionalidad de proceso de la aplicación. En general, la seguridad se activará para casi todos los métodos de proceso. La seguridad se puede desactivar para un método de proceso que no necesite la seguridad, por ejemplo, un método de inicio de sesión que se invoque cuando un usuario intente iniciar la sesión en una aplicación. Mientas un usuario aún no se haya autenticado ni autorizado, necesitará acceder a este método a fin de iniciar la sesión, por lo que desactivar la seguridad de este método puede resultar necesario.

Durante la fase de diseño inicial de una aplicación, la sobrecarga de mantener el entorno de seguridad "en sincronización" con una aplicación en evolución puede resultar tedioso. Se puede inhabilitar la comprobación de autorización estableciendo la propiedad curam.security.disable.authorisation en el archivo Application.prx.

advertencia: Advertencia

La propiedad curam.security.disable.authorisation sólo debe activarse en la fase de diseño. No debe establecerse nunca en `true` en un entorno de producción.

Finalmente, debe tener en cuenta que una vez el código y los scripts se han generado a partir del modelo de trabajo, no se puede cambiar la información asociada con un FID. Cambiar esta información requiere una modificación del modelo y regenerar y recompilar la base de datos.

11.13 Control del registro de errores de autorización del cliente

De forma predeterminada, no se registran los errores de autorización del cliente web.

La propiedad curam.enable.logging.client.authcheck controla si los errores de autorización encontrados por el cliente web son registrados o no. Esta propiedad es`false` de forma predeterminada, lo que significa que estos errores no se registran. Cuando se establece en `true`, se almacena un registro de estos errores de autorización en la tabla de base de datos AuthorisationLog. Puede consultar la publicación *Cúram Server Developers Guide* (Guía de desarrolladores del servidor de Cúram), en la sección Application.prx - Propiedades dinámicas, para obtener más información sobre esta propiedad.

11.14 Autorización de Nuevos tipos de SID (identificador de seguridad)

Se suministra un método de interfaz de servidor para habilitar la autorización que debe realizarse directamente. Este método puede añadirse a una clase que manipule datos en el elemento conceptual que está protegido por el nuevo tipo de SID (identificador de seguridad).

```
curam.util.security.Authorisation.isSIDAuthorised()
```

A continuación encontrará un ejemplo de uso de este método:

```
// El SID asociado con el elemento conceptual
       // que debe protegerse.
       String someSID = "someSID";

       // Obtener el nombre de usuario que ha iniciado la sesión
       String loggedUser =
         curam.util.transaction.TransactionInfo.getProgramUser();

       // Comprobar si el usuario dispone de derechos de acceso
       if (curam.util.security.Authorisation.isSIDAuthorised(
             someSID, loggedUser)) {
         // Hacer algo sensible para lo que el usuario dispone de derechos
         ...
       } else {
       // Emitir una excepción que indique que el usuario no dispone de
       // acceso para realizar esta acción
         AppException exception
           = new AppException(MESSAGE.ERR_USER_NO_ACCESS);
         throw exception;
       }
```

Figura 7. Ejemplo de uso de isSIDAuthorised()

11.15 Análisis de la tabla de base de datos AuthorisationLog

Todos los errores de autenticación se registran en una tabla de base de datos llamada AuthenticationLog. A continuación encontrará las filas que resultan de interés en esta tabla:

Tabla 2. Contenido del registro de autenticación

Campo	**Significado**
timeEntered	La indicación de fecha y hora de la entrada en el registro.
userName	El nombre de usuario asociado con el intento de autorización.
identifierName	El identificador de seguridad (SID) o identificador funcional (FID) asociados al error.

El API `LogAdmin` puede utilizarse para consultar la tabla de base de datos AuthorisationLog. Consulte la documentación de Java de esta clase para obtener más detalles.

Capítulo 12. Personalización de la criptografía

12.1 Descripción general

Este capítulo muestra en detalle cómo configurar y personalizar la criptografía de IBM Cúram Social Program Management.

12.2 Personalización del cifrado

La modificación de los valores de cifrado predeterminados es un proceso relativamente simple, pero requiere una planificación y unas pruebas adecuados. Será necesario reiniciar la aplicación para que los cambios entren en vigor y, dependiendo del tamaño y de la topología de la organización y de los despliegues, será necesario elegir un momento en el que los cambios en curso no tengan impacto. Asimismo, tenga en cuenta aquellos datos (p.ej. propiedades que contengan contraseñas encriptadas) gestionados por Cúram Transport Manager (CTM) que tengan que actualizarse o gestionarse para evitar que los sistemas dejen de estar sincronizados entre sí (consulte la *Guía de Cúram Transport Manager* para obtener información adicional).

La modificación de los valores de cifrado predeterminados conlleva los pasos siguientes:

1. Elegir nuevos valores para `CryptoConfig.properties` y los artefactos subyacentes - consulte 4.5, “Valores de cifrado de Cúram”, en la página 22
2. En función de los valores, puede que tenga que realizar pasos adicionales (por ejemplo, cuando se modifica el almacén de claves como se indica en 12.4, “Cómo crear un almacén de claves”, en la página 48).
3. Modificar el archivo `CryptoConfig.properties`; tenga en cuenta que la ubicación predeterminada es `<DIR_SERVIDOR>/project/properties`.
4. Eliminar los archivos `CryptoConfig.jar` existentes (contienen `CryptoConfig.properties`), que se encuentran en el `directorio <JAVA_HOME>/jre/lib/ext` (`$JAVA_HOME/lib/ext` en IBM z/OS). Si hay algún cliente o servidor de Cúram en ejecución, deberá pararse para poder desplegar un archivo `CryptoConfig.jar` con los valores actualizados.
5. Volver a encriptar las contraseñas en todos los archivos de propiedades identificados en 4.7, “Contraseñas cifradas”, en la página 24. Los destinos de Apache Ant configtest, configure e installapp colocarán un archivo `CryptoConfig.jar` actualizado en el directorio `lib/ext` de Java.
6. Probar y verificar los cambios.

La prueba de los cambios debería incluir la verificación de la funcionalidad afectada; por ejemplo:

- Comprobar que el destino configtest de Ant sigue funcionando.
- Comprobar que los programas por lotes siguen funcionando.
- Cuando proceda, comprobar que el destino Ant configure sigue funcionando.

Temas relacionados:

- 4.6, “Valores de resumen de Cúram”, en la página 23
- 4.7, “Contraseñas cifradas”, en la página 24

12.3 Gestión de claves

La gestión de la clave secreta de las contraseñas encriptadas de Cúram se realiza mediante el comando **`keytool`** proporcionado por el JDK, o mediante uno similar. Deberán tomarse decisiones locales relativas a la ubicación y al aislamiento de la clave secreta de Cúram que sean compatibles con la organización y los estándares locales.

© Copyright IBM Corp. 2012, 2013

Tenga en cuenta que algunos valores pasados al comando **keytool** deben reflejarse en los valores de CryptoConfig.properties, lo que requiere una coordinación para lograr un despliegue satisfactorio tal y como se expone en 12.2, "Personalización del cifrado", en la página 47. La tabla siguiente muestra la relación entre los argumentos del comando **keytool** y las propiedades criptográficas de Cúram.

Tabla 3. Relación de los argumentos del comando keytool con las propiedades criptográficas de Cúram

Argumento de keytool	propiedad CryptoConfig.properties
-keyalg	curam.security.crypto.cipher.algorithm
-alias	curam.security.crypto.cipher.keystore.seckey.alias
-keystore	curam.security.crypto.cipher.keystore.location
-storepass	curam.security.crypto.cipher.keystore.storepass

Nota: La contraseña de la clave secreta toma el valor predeterminado de storepass y no debería cambiarse.

Consulte la documentación del JDK para obtener más información sobre cómo utilizar el comando **keytool** .

Temas relacionados:

- 4.5, "Valores de cifrado de Cúram", en la página 22
- 4.4, "Propiedades de criptografía", en la página 22
- 12.4, "Cómo crear un almacén de claves"

12.4 Cómo crear un almacén de claves

La creación de un almacén de claves que sustituya el almacén predeterminado de Cúram requiere ejecutar el comando **keytool** que se proporciona en el JDK (o uno equivalente), modificar los correspondientes valores de CryptoConfig.properties (solo es necesario si el nombre y/o la ubicación del almacén de claves son distintos de los predeterminados, aunque el cambio del nombre puede hacer que las personalizaciones sean más evidentes) y garantizar que los destinos Ant de Cúram puedan encontrar el nuevo almacén de claves (solo es necesario si cambia la ubicación predeterminada).

Por ejemplo:

```
keytool -genseckey -v  -alias MiClaveSecreta -keyalg AES -keysize 128 -keystore MyOrganizacion.keystore -storepass contrase
```

La sección 12.3, "Gestión de claves", en la página 47 identifica los argumentos del comando **keytool** relacionados con los valores de CryptoConfig.properties.

La ubicación predeterminada del archivo del almacén de claves es el directorio <DIR_SERVIDOR>/project/properties directorio con una estructura de subdirectorios que refleja el JDK en uso: "ibm" para el JDK de IBM y "sun" para el JDK de Oracle. Así, cuando se crea un archivo de almacén de claves, los scripts de compilación de Cúram esperarán encontrarlo en <DIR_SERVIDOR>/project/properties/ibm (en el caso del JDK de IBM). Si desea utilizar una ubicación distinta de la predeterminada, puede hacer una de estas dos cosas:

1. Utilizar una ubicación absoluta para el archivo de almacén de claves tal y como se describe en 4.4, "Propiedades de criptografía", en la página 22. En este caso, se ignorarán los archivos de almacén de claves predeterminados de Cúram en CryptoConfig.jar y se tendrá en cuenta el valor absoluto de CryptoConfig.properties.
2. Utilizar la propiedad de Ant crypto.prop.file.location cuando se ejecute cualquiera de los destinos (descritos en 12.2, "Personalización del cifrado", en la página 47) que crean y copian CryptoConfig.jar para que apunten a la ubicación alternativa. La ubicación especificada tendrá que reflejar la estructura del JDK: "ibm" o "sun". Por ejemplo:

- coloque el nuevo archivo de almacén de claves en una ubicación como esta (para el JDK de IBM): `C:\Curam\keystore\ibm\MiOrganizacion.keystore`
- apunte a dicha ubicación cuando ejecute los destinos de compilación: `ant configure -Dcrypto.prop.file.location=C:\Curam\keystore`

Nota: En el ejemplo anterior, el cambio de nombre del archivo del almacén de claves a `MiOrganizacion.keystore` requerirá el correspondiente cambio en `CryptoConfig.properties`, como se indica en 4.4, “Propiedades de criptografía”, en la página 22.

Una vez creado el almacén de claves, deberá seguir los pasos indicados en 12.2, “Personalización del cifrado”, en la página 47.

Temas relacionados:

- 12.3, “Gestión de claves”, en la página 47
- 12.2, “Personalización del cifrado”, en la página 47

12.5 Personalización de un resumen

La modificación de los valores de resumen predeterminados es un proceso relativamente simple, pero requiere una planificación y unas pruebas adecuados. Será necesario reiniciar la aplicación para que los cambios entren en vigor y, dependiendo del tamaño y de la topología de la organización y de los despliegues, será necesario elegir un momento en el que los cambios en curso no tengan impacto. Asimismo, tenga en cuenta aquellos datos (p.ej. contraseñas de usuario) gestionados por Cúram Transport Manager (CTM) que tengan que actualizarse o gestionarse para evitar que los sistemas dejen de estar sincronizados entre sí (consulte la *Guía de Cúram Transport Manager* para obtener información adicional).

El proceso se trata en detalle en 12.7, “Cómo utilizar los valores de resumen sustituidos durante un período de migración”, en la página 50.

Temas relacionados:

- 4.6, “Valores de resumen de Cúram”, en la página 23
- 12.6, “Cómo especificar una sal de resumen”

12.6 Cómo especificar una sal de resumen

Si bien Cúram no la especifica de forma predeterminada, la especificación de una sal para contraseñas resumidas proporciona un nivel adicional de protección frente a ataques de fuerza bruta.

Para especificar una sal para las contraseñas resumidas:

1. Elija una cadena aleatoria lo suficientemente larga.
2. Encripte dicha cadena mediante el destino Ant **`encrypt`** (tal y como se documenta en la *Guía del desarrollador de Cúram Server*.
3. Guarde la cadena encriptada en un archivo.
4. Especifique la ubicación del archivo que contiene la cadena de sal encriptada utilizando la propiedad `curam.security.crypto.digest.salt.location` en `CryptoConfig.properties` y asegúrese de que todos los archivos `CryptoConfig.jar` desplegados reflejen los valores actualizados.

A fin de que el proceso sea manejable, dichos cambios deberían realizarse junto con los pasos indicados en 12.7, “Cómo utilizar los valores de resumen sustituidos durante un período de migración”, en la página 50.

12.7 Cómo utilizar los valores de resumen sustituidos durante un período de migración

La utilización de valores de resumen reemplazados significa que se están migrando las contraseñas resumidas existentes a una nueva configuración criptográfica (p.ej. nueva sal) y se desea que las contraseñas de usuario de Cúram se migren automáticamente durante un período de tiempo. Esto se aplica a los usuarios de Cúram externos e internos, pero no se aplica a los usuarios gestionados por sistemas de seguridad de terceros como LDAP.

El proceso para hacer esto es:

1. Elija un momento en el que el sistema de Cúram pueda estar parado, es decir, que el sistema Cúram no ejecute.
2. Copie los nombres y valores de las propiedades de resumen existentes en `CryptoConfig.properties` y renombre las propiedades a los nuevos nombres de las propiedades reemplazadas.
3. Modifique los nombres de propiedad de resumen existentes en `CryptoConfig.properties`.
4. Establezca la propiedad `curam.security.convertsupersededpassworddigests.enabled` a `'true'`.
5. Defina la propiedad `curam.security.crypto.upgrade.start` para que le ayude a realizar un seguimiento del momento en que se introduce la configuración actualizada. Este valor puede utilizarse abajo para ayudar a gestionar las contraseñas de usuario no migradas.
6. Reinicie el servidor de aplicaciones, pero tenga en cuenta lo siguiente:

Nota: Ni el usuario predeterminado de servicios web de Cúram (WEBSVCS) ni cualquier usuario no procesado vía `CuramLoginModule` estarán disponibles para la migración automática de contraseñas. Dichos usuarios deberán restablecerse antes de reiniciar el servidor de aplicaciones. Para hacer esto:

1. Obtenga el nuevo valor de resumen de contraseña mediante el destino Ant digest (p.ej. `ant digest -Dpassword=contrasenya`).
2. Actualice el valor de la contraseña en la base de datos, lo que puede hacerse fácilmente mediante SQL (p.ej. `UPDATE USERS SET PASSWORD='<nuevo valor de resumen>' WHERE USERNAME='WEBSVCS';`).
3. Ahora ya se puede iniciar el servidor de aplicaciones.

Transcurrido un período de tiempo (p.ej. semanas o meses), cuando se considere que el período de migración ha concluido, establezca la propiedad `curam.security.convertsupersededpassworddigests.enabled` a `'false'` y desestablezca la `propiedad curam.security.crypto.upgrade.start`.

A aquellos usuarios que no hayan iniciado sesión durante el período de migración les fallará el inicio de sesión al no coincidir las contraseñas. Existen dos formas de abordar el problema de las contraseñas no actualizadas durante el período de migración:

1. Solicite a dichos usuarios que se pongan en contacto con el soporte interno para que se restablezcan sus contraseñas a través de la interfaz del usuario administrador.
2. Identifique manualmente en la tabla USERS aquellos usuarios que no se actualizaron durante el período de migración y establezca la nueva contraseña predeterminada vía SQL (consulte el destino **`digest`** descrito en la *Guía del desarrollador de Cúram Server* para obtener nuevos valores de resumen de contraseña) o bien a través de las pantallas del usuario administrador. Por ejemplo, mediante la consulta siguiente: `SELECT nombreusuario FROM users WHERE lastwritten between timestamp('2013-06-01 15:00:00') AND timestamp('2013-09-01 00:00:00')`

La propiedad `curam.security.convertsupersededpassworddigests.enabled` no debería dejarse a true indefinidamente porque:

1. no tiene sentido tomarse la molestia de actualizar de la configuración 'A' a la configuración 'B' y dejar la configuración 'A' activa;
2. deja en el sistema valores criptográficos potencialmente más débiles; y

3. para poder utilizar esta funcionalidad en una actualización futura, p.ej. de la configuración 'B' a la 'C', habría que actualizar todas las contraseñas de 'A' a al menos 'B'.

Nota: En lo relativo a la estrategia de migración, deberán tenerse en cuenta todos los archivos que tengan resúmenes almacenados (p.ej. DMX) a fin de que reflejen los valores correctos.

Nota: Cualquier uso que se haga de Cúram Transport Manager (CTM) durante una migración deberá estudiarse a fin de garantizar las expectativas y los valores compatibles entre los sistemas origen y destino.

Temas relacionados:

- 4.5, “Valores de cifrado de Cúram”, en la página 22
- 4.6, “Valores de resumen de Cúram”, en la página 23

12.8 Modificación de la configuración criptográfica en un sistema de producción

Si bien los valores criptográficos que van predefinidos en el producto (OOTB) son válidos para un entorno de desarrollo o de pruebas, deberían cambiarse en un entorno de producción para dar protección y aislar un entorno de producción de alto riesgo de dichos entornos de nivel de riesgo relativamente bajo.

Entre los cambios más habituales que pueden efectuarse a la configuración criptográfica OOTB cabe mencionar:

- Proporcionar una nueva clave secreta.
  - Dicha clave puede generarse con el programa de utilidad keytool del JDK; consulte 12.4, “Cómo crear un almacén de claves”, en la página 48
    - Esta clave secreta debería almacenarse en un almacén de claves aparte.
    - Las propiedades de estos cambios de clave secreta son las descritas en 12.3, “Gestión de claves”, en la página 47.
- Proporcionar nuevos valores de resumen.
  - Los nuevos valores de resumen pueden incluir una sal, un recuento de iteraciones y/o un algoritmo nuevos.
    - Las propiedades de estos cambios de resumen son las descritas en 4.6, “Valores de resumen de Cúram”, en la página 23 y 12.6, “Cómo especificar una sal de resumen”, en la página 49, y el proceso es el descrito en 12.7, “Cómo utilizar los valores de resumen sustituidos durante un período de migración”, en la página 50.

No olvide mantener los archivos de configuración aislados de todo aquel que no tenga una necesidad absoluta de acceder a ellos; en concreto, mantenga aislada la información de configuración entre los entornos de desarrollo, pruebas y producción.

Capítulo 13. Personalización de aplicaciones de usuario externo

13.1 Visión general

Puesto que los usuarios externos se procesan de forma distinta que los usuarios internos, se requiere de forma específica una aplicación web de IBM Cúram Social Program Management independiente para usuarios externos.

13.2 Creación de una aplicación de usuario externo

Debe desplegarse una nueva aplicación de cliente web para usuarios externos. Debería consultar la publicación *Cúram Web Client Reference Manual* (Manual de consulta de cliente web de Cúram) para obtener detalles sobre cómo crear una nueva aplicación de cliente web.

13.3 Creación de una página de inicio de sesión de usuario externo

Debe crearse un nuevo logon.jsp para una aplicación de usuario externo. La plataforma Social Program Management incluye una página de inicio de sesión predeterminada, `logon.jsp`, ubicada en el directorio `lib/curam/web/jsp` de CDEJ (Client Development Environment for Java). Este archivo debe copiarse en una carpeta `webclient/components/<custom component>/WebContent` en la aplicación de cliente web y modificarse de la siguiente manera:

El elemento `table` debe ampliarse para que incluya un campo de entrada oculto user_type:

```
<input type="hidden" name="user_type"
          value="EXTERNAL"/>
```

donde `EXTERNAL` indica el tipo de usuario externo. Se puede establecer en cualquier valor menos `INTERNAL`.

13.4 Creación de una página de inicio de sesión automática de cliente de usuario externo

Algunas aplicaciones cliente de usuario externo no requieren autorización de usuario y, por ello, no debe solicitarse ni el nombre de usuario ni la contraseña. No se puede inhabilitar la autenticación en IBM Cúram Social Program Management, por lo que la mejor forma de cumplir con este requisito es escribir un script de inicio de sesión automático.

El script de inicio de sesión automático adopta un nombre de usuario y contraseña codificados y los facilita como información de autenticación cuando así se le solicita. Ello significa que todos los usuarios para tal aplicación siempre se ejecutarán bajo el mismo nombre de usuario. La utilización de tal script debe limitarse a ciertas aplicaciones de acceso abierto.

Al implementar aplicaciones que necesitan un inicio de sesión automático, deben considerarse las implicaciones para la gestión de sesiones. La gestión de sesiones en IBM Cúram Social Program Management mantiene información de la sesión del usuario para garantizar que cuando el usuario vuelva a iniciar la sesión, se abra la información de la sesión relevante, es decir, sus pestañas y navegación, en el mismo sitio donde las dejó. En el caso de un usuario que haya iniciado la sesión automáticamente, esta información no debe conservarse y, por ello, es posible que deba desactivarse la gestión de sesiones en este escenario. Debe consultar la publicación *Cúram Web Client Reference Manual* (Manual de consulta del cliente web de Cúram) para obtener más detalles sobre cómo realizar la desactivación.

© Copyright IBM Corp. 2012, 2013

A continuación encontrará ejemplos de scripts JSP de inicio y finalización de sesión automático.

Nota: Las implementaciones de seguridad y las configuraciones difieren entre los proveedores del servidor de aplicaciones por lo que es posible que estos ejemplos no funcionen en todos los caso o para todas las versiones del servidor de aplicaciones.

```
<?xml version="1.0" encoding="UTF-8"?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
  xmlns:prefix="URI"
  version="2.0">
  <jsp:directive.page buffer="32kb"
                      contentType="text/html; charset=UTF-8"
                      pageEncoding="UTF-8" />
  <jsp:text>
    <![CDATA[
      <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">]]>
  </jsp:text>

  <!-- Automatic redirect to login security check of user
          details specified below -->

  <html>
    <head>
      <script type="text/javascript">
        function autoSubmit() {
          document.getElementById("loginform").submit();
        }
      </script>
      <meta content="text/html; charset=UTF-8"
            http-equiv="Content-Type" />
    </head>
    <body class="logonBody"
          style="visibility: hidden;"
          onload="autoSubmit()">
      <form id="loginform"
            name="loginform"
            action="j_security_check"
            method="post">
        <input type="hidden"
               name="j_username"
               value="generalpublic" />
        <input type="hidden"
               name="j_password"
               value="password" />
        <input type="hidden"
               name="user_type"
               value="EXTERNAL" />
      </form>
    </body>
  </html>
</jsp:root>
```

Figura 8. JSP de inicio de sesión automático

```
<?xml version="1.0" encoding="UTF-8"?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
  xmlns:prefix="URI"
  version="2.0">
  <jsp:directive.page buffer="32kb"
                      contentType="text/html; charset=UTF-8"
                      pageEncoding="UTF-8" />
  <jsp:text>
    <![CDATA[
      <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">]]>
  </jsp:text>
  <html>
    <head>
      <script type="text/javascript">
        function autoSubmit() {
          document.getElementById("logout").submit();
        }
      </script>
      <meta content="text/html; charset=UTF-8"
            http-equiv="Content-Type" />
    </head>
    <body class="logoutBody"
          style="visibility: hidden;"
          onload="autoSubmit()">
      <form id="logout"
            name="logout"
            action="servlet/ApplicationController"
            method="post">
        <input type="submit"
               name="j_logout"
               value="Log Out" />
        <input type="hidden"
               name="logoutExitPage"
               value="redirect.jsp" />
      </form>
    </body>
  </html>
</jsp:root>
```

Figura 9. JSP de finalización de sesión automática

13.5 Extensión de la clase de usuario de acceso público

Para “enganchar” la solución personalizada a la aplicación, deberá extenderse la clase abstracta `curam.util.security.PublicAccessUser`, lo que requiere implementar la interfaz `curam.util.security.ExternalAccessSecurity`. Esa clase concreta se utilizará durante el proceso de autorización y autenticación para determinar la información necesaria relativa al usuario externo. Esta clase y sus métodos se describen a continuación de forma detallada.

13.6 Autenticación de un usuario externo

El método `authenticate()` es el responsable de autenticar a un usuario externo. Se invoca durante el proceso de autenticación si el usuario está identificado como un usuario externo. En el caso de usuarios externos, este método se invoca en lugar de la autenticación configurada.

Nota: Si se ha configurado un mecanismo de autenticación alternativo, por ejemplo LDAP, los usuarios externos deben poder autenticarse ante este mecanismo.

```
/**
   * La implementación de este método debería validar el identificador y la
   * contraseña y devolver el resultado de la validación. Si la información es
   * válida, debería devolverse el código de tabla de códigos SecurityStatus.LOGIN.
   *
   * @param identifier El identificador del usuario externo.
   * @param password La contraseña como un vector de caracteres.
   * @param userType El tipo de usuario externo.
   *
   * @return El estado de la autenticación en forma de un código de tabla de códigos.
   *
   * @throws AppException Firma de excepción genérica.
   * @throws InformationalException Firma de excepción genérica.
   */

  public abstract String authenticate(String identifier,
    char[] password, String userType)
    throws AppException, InformationalException;
```

Los parámetros de entrada del método incluyen un identificador, un resumen (digest) de la contraseña en forma de vector de caracteres y el tipo de usuario externo que debe autenticarse.

El parámetro `userType` está previsto para soportar varios tipos de usuarios externos que requieran distintos mecanismos de autenticación. El usuario de este parámetro depende de la implementación personalizada.

El resultado esperado de este método será una entrada de la tabla de códigos `curam.util.codetable.SECURITYSTATUS`. Si el resultado de la autenticación es satisfactorio, el resultado debe se:

```
curam.util.codetable.SECURITYSTATUS.LOGIN
```

Para los errores de autenticación, esta tabla de códigos contiene numerosas entradas que incluyen `BADUSER`, `BADPWD` y `PWDEXPIRED`. Esta tabla de códigos se puede ampliar para que incluya códigos personalizados tal como se explica en la *Guía del desarrollador de Cúram Server*.

El resultado de la autenticación devuelto por este método se registra automáticamente en la tabla de base de datos `AuthenticationLog` . Para obtener más información sobre esta tabla, consulte la publicación *Cúram Server Developers Guide* (Guía de los desarrolladores del servidor de Cúram).

La clase abstracta `PublicAccessUser` también define los siguientes métodos abstractos que debe implementar cualquier subclase concreta:

- El método `upgradeSafePasswordValidation()` es necesario para comparar contraseñas y se define de la manera siguiente:

```
public final boolean upgradeSafePasswordValidation(
final String nombreUsuario,
final String hashContrasenyaAlmacenado,
final String contrasenyaEnClaro)
```

- El método `setPassword()` permite al implementador persistir la contraseña (p.ej. una contraseña nueva) en el caso de actualizaciones criptográficas. Por tanto, este método se llama cuando se llama el método `upgradeSafePasswordValidation()`. Esta es la definición del método:

```
public abstract void setPassword(String nombreUsuario, String hashContrasenya)
throws AppException, InformationalException;
```

Consulte la correspondiente JavaDoc de la clase `PublicAccessUser` para obtener detalles adicionales sobre los métodos anteriores.

13.7 Determinación de detalles de usuario externo

Los detalles de un usuario externo se recuperan llamando al método `getLoginDetails()` de la interfaz `curam.util.security.ExternalAccessSecurity`. Estos detalles se devuelven directamente después de la autenticación para dirigir al usuario externo a la página de inicio correcta de la aplicación.

```
/**
 * La implementación de este método debería recuperar los
 * detalles del usuario necesario para redireccionarlos a la página
 * de la aplicación correcta. Esta información incluye el nombre de la
 * página de inicio de la aplicación del usuario, el entorno local predeterminado del
 * usuario y una lista de advertencias/mensajes para el usuario.
 *
 * @param identifier El identificador del usuario externo.
 *
 * @return Los detalles del usuario, incluyendo la página de inicio de la
 *         aplicación.
 *
 * @throws AppException Firma de excepción genérica.
 * @throws InformationalException Firma de excepción genérica.
 */
UserLoginDetails getLoginDetails(identificador de serie)
  throws AppException, InformationalException;
```

Debe crearse una instancia de `curam.util.security.UserLoginDetails` y debe ser devuelta por este método. Debería devolverse esta información utilizando esta clase:

- `UserLoginDetails. setApplicationCode(String code)`

  El código que corresponde a la página de inicio de la aplicación para el usuario externo.

  Debe ser una entrada válida en la tabla de códigos `APPLICATION_CODE`.
- `UserLoginDetails. setDefaultLocale(String defaultLocale)`

  El entorno local predeterminado para el usuario externo.

  Se trata del entorno local en el que se visualizará la aplicación para el usuario externo.
- `UserLoginDetails. addInformationals(InformationalManager informationalManager)`

  Cualquier tipo de mensaje informativo que deba aparecer al usuario externo.

  La clase `curam.util.exception.InformationalManager` se puede utilizar para crear numerosos mensajes informativos o de advertencia que se visualizarán cuando el usuario externo inicie la sesión. Por ejemplo, una advertencia para permitir que el usuario externo sepa que su contraseña va a caducar.

13.8 Autorización de un usuario externo

El método `getSecurityRole()` se utiliza durante la autorización para determinar el rol de seguridad asociado con el usuario externo. Los roles de seguridad utilizados para usuarios externos se configuran de la misma manera que los roles de seguridad para usuarios internos.

```
/**
 * La implementación de este método debería devolver el rol de
 * seguridad asociado con el usuario externo para poder realizar la
 * autorización. Si el usuario no existe, debería devolver
 * nulo.
 *
 * @param identifier El identificador del usuario externo.
 *
 * @return El rol de seguridad de la autorización.
 *
 * @throws AppException Firma de excepción genérica.
 * @throws InformationalException Firma de excepción genérica.
 */
String getSecurityRole(identificador de serie)
  throws AppException, InformationalException;
```

El SDEJ invocará una implementación de esté método durante el proceso de autorización si el usuario no existe en la memoria caché de seguridad. Sólo pueden existir usuarios internos en la memoria caché de seguridad. Ello significa que los identificares utilizados para identificar usuarios externos deben ser exclusivos y no entrar en conflicto con la configuración de nombres de usuario de usuarios internos, a menos que se haya implementado la interfaz `UserScope` personalizada tal como se describe en 7.3, “Ámbito del usuario”, en la página 31. De lo contrario, si cualquier nombre de usuario entra en conflicto con derechos de acceso asignados al usuario interno, también se utilizará para el usuario externo.

Si no se puede determinar un rol para el usuario externo, deberá devolver nulo de forma que el SDEJ pueda notificar el error de autorización correctamente.

13.9 Determinación del tipo de usuario

El método `getUserType()` se utiliza para determinar si un usuario es un usuario externo.

```
/**
 * Devuelve el tipo de usuario. Es para permitir el soporte de
 * distintos tipos de usuario externo. Si sólo hay un
 * tipo de usuario externo, simplemente devuelve "EXTERNAL".
 *
 * @param identifier El identificador del usuario externo.
 *
 * @return El tipo de usuario externo.
 *
 * @throws AppException Firma de excepción genérica.
 * @throws InformationalException Firma de excepción genérica.
 */
 String getUserType(identificador de serie final)
   throws AppException, InformationalException;
```

El `getProgramUserType()` en `curam.util.transaction.TransactionInfo` invocará a este método para que devuelva el tipo de usuario si el usuario no se reconoce como un usuario interno. Para los usuarios internos, siempre se devuelve “INTERNAL”.

Para los usuarios externos, hay varios tipos de usuarios externos, por lo que este método debe devolver el tipo específico de usuario externo.

13.10 Cómo impedir la supresión de un rol de seguridad: cuento de uso de rol

El método `getRoleUsageCount()` se utiliza para impedir la supresión de un rol de seguridad que actualmente sea referenciado por un usuario externo.

```
/**
 * Devuelve el número de usuarios que utilizan un rol en concreto. Este
 * método se utiliza para garantizar que no se pueda suprimir un rol cuando
 * lo esté utilizando un usuario externo.
 *
 * @param role El nombre del rol de seguridad.
 *
 * @return El número de usuarios que actualmente están utilizando el
 *         rol especificado.
 *
 * @throws AppException Firma de excepción genérica.
 * @throws InformationalException Firma de excepción genérica.
 */
 int getRoleUsageCount(String role)
   throws AppException, InformationalException;
```

Los roles de seguridad que son referenciados por cualquier usuario, interno o externo, no se pueden eliminar. Este método debe devolver un número de 1 o más si cualquier de los usuarios externos hace referencia al rol especificado.

13.11 Recuperación de un nombre de usuario registrado

El método `getRegisteredUserName()` se utiliza para recuperar el nombre de usuario con caso correcto, lo que puede ser independiente del nombre de usuario escrito durante el inicio de sesión.

```
/**
 * Obtiene el caso correcto de este usuario independientemente del caso
 * mixto que se haya podido escribir por parte del usuario que ha iniciado la sesión.
 *
 * @param identifier El identificador del usuario externo,
 * cuyo caso puede que no coincida con el del identificador guardado
 * para el usuario.
 *
 * @return El caso real del usuario, antes de que su caso haya sido modificado por
 * factores externos.
 *
 * @throws AppException Firma de excepción genérica.
 * @throws InformationalException Firma de excepción genérica.
 */
 public String getRegisteredUserName(identificador de serie final)
   throws AppException, InformationalException;
```

La implementación predeterminada de este método debería devolver el nombre de usuario que se ha suministrado. Sólo si se ha establecido curam.security.casesensitive en false, este método podría tener que cambiar el caso del nombre de usuario devuelto.

Nota: Cuando la propiedad curam.security.casesensitive se ha establecido en false y es necesaria para usuarios externos, es responsabilidad de todos los métodos de esta interfaz el manejar todos los requisitos específicos de caso.

13.12 Lectura de preferencias de usuario

El método `getUserPreferenceSetID()` se utiliza para recuperar el ID de conjuntos de preferencias de usuario asociado con un usuario externo. Si no existen preferencias de usuario para un usuario externo, entonces se utilizan las preferencias predeterminadas para el usuario externo. El capítulo *Preferencias de usuario* de la publicación *Cúram Server Developer's Guide* (Guía del desarrollador del servidor de Cúram) proporciona más detalles sobre las preferencias de usuario.

```
/**
 * Este método se utiliza para recuperar un conjunto de preferencias de usuario
 * asociado con un usuario externo. El userPrefSetID es una
 * clave foránea para la tabla UserPreferenceInfo.
 * La tabla UserPreferenceInfo contiene información sobre
 * las preferencias de usuario.
 *
 * @param identifier El identificador del usuario externo.
 *
 * @return El userPrefSetID del usuario externo.
 *
 * @throws AppException Firma de excepción genérica.
 * @throws InformationalException Firma de excepción genérica.
 */
 String getUserPreferenceSetID(identificador de serie final)
   throws AppException, InformationalException;
```

La implementación predeterminada de este método debe devolver el ID del conjunto de preferencias de usuario para las preferencias de usuario asociadas con un usuario externo.

13.13 Modificación de preferencias de usuario

El método `modifyUserPreferenceSetID()` se utiliza para actualizar los detalles del usuario externo con un nuevo conjunto de preferencias de usuario. Consulte las preferencias de usuario para obtener más información sobre las preferencias de usuario.

```
/**
 * Este método actualiza los detalles del usuario externo con nuevas preferencias de
 * usuario.
 *
 * @param userPreferenceSetID El ID de las preferencias de usuario.
 * @param username El identificador del usuario externo.
 *
 * @throws AppException Firma de excepción genérica.
 * @throws InformationalException Firma de excepción genérica.
 */
 void modifyUserPreferenceSetID(
  final String userPreferenceSetID, nombre de usuario de serie final)
     throws AppException, InformationalException;
```

La implementación predeterminada de este método debe actualizar el ID del conjunto de preferencias de usuario asociado con un usuario externo.

13.14 Configuración de la seguridad de acceso externo

La propiedad `curam.custom.externalaccess.implementation` debe estar establecida en `Application.prx` para indicar el nombre completo de la clase que implementa la interfaz anterior.

Nota: La propiedad curam.custom.externalaccess.implementation no es dinámica y si se modifica, la aplicación debe reiniciarse antes de que el cambio entre en vigor.

13.15 Determinación de si un usuario es interno o externo utilizando la interfaz UserScope

Para dar soporte a métodos alternativos para determinar si un usuario es interno o externo, dispone de la interfaz personalizada `UserScope`. Por ejemplo, esta interfaz personalizada puede implementarse para determinar el tipo de usuario en base a información adicional y eliminar el requisito de nombres exclusivos entre usuarios internos y externos.

Para proporcionar una implementación personalizada para determinar el tipo de usuario, debe implementarse la interfaz `curam.util.security.UserScope`. Esta interfaz tiene un método `isUserExternal()` que determina el tipo de usuario. Este método debería devolver true si el usuario se considera externo o false si el usuario es interno.

Para especificar la implementación personalizada que debe utilizarse, la propiedad curam.custom.userscope.implementation debe estar establecida en `Application.prx`. Así se establecería el nombre completo de la clase que implementa la interfaz `UserScope`.

Nota: La propiedad curam.custom.userscope.implementation no es dinámica y si se modifica, la aplicación debe reiniciarse antes de que el cambio entre en vigor.

El método isUserExternal() de la interfaz UserScope se detalla a continuación:

13.16 Determinación de tipo de usuario

Se invoca el método `isUserExternal()` en cualquier parte de la aplicación en la que deba determinarse el tipo de usuario. Ello incluye el usuario que inicia la sesión en la aplicación y cuando prueba la autorización para acceder a elementos seguros de IBM Cúram Social Program Management.

```
/**
 * La implementación de este método debería determinar el tipo de
 * usuario que ha iniciado la sesión en la aplicación. Hay 2 tipos de
 * usuario: INTERNAL y EXTERNAL. Si el usuario es un usuario EXTERNAL,
 * entonces este método debería devolver true. Si devuelve false,
 * a continuación el usuario se considera INTERNAL.
 *
 * @param username - El nombre de usuario.
 * @return Un valor booleano de true que indica un usuario EXTERNAL,
 * false indica un usuario INTERNAL.
 *
 * @throws AppException Firma de excepción genérica.
 * @throws InformationalException Firma de excepción genérica.
 */
boolean isUserExternal(Serie de nombre de usuario)
  emite AppException, InformationalException;
```

Avisos

Esta información se ha desarrollado para productos y servicios ofrecidos en los Estados Unidos. Es posible que IBM no ofrezca los productos, servicios o características que se describen en este documento en otros países. Solicite información al representante local de IBM acerca de los productos y servicios disponibles actualmente en su zona. Cualquier referencia a un producto, programa o servicio de IBM no pretende afirmar ni implica que sólo pueda utilizarse ese producto, programa o servicio de IBM. En su lugar, se puede utilizar cualquier producto, programa o servicio funcionalmente equivalente que no vulnere ningún derecho de propiedad intelectual de IBM. No obstante, es responsabilidad del usuario evaluar y verificar el funcionamiento de cualquier producto, programa o servicio que no sea de IBM. IBM podría tener patentes o solicitudes de patentes pendientes relacionadas con el tema principal que se describe en este documento. La posesión de este documento no confiere ninguna licencia sobre dichas patentes. Puede enviar consultas sobre licencias, por escrito, a:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

EE.UU.

Para consultas sobre licencias relacionadas con información de doble byte (DBCS), póngase en contacto con el departamento de propiedad intelectual de IBM de su país o envíe sus consultas, por escrito, a:

Intellectual Property Licensing

Legal and Intellectual Property Law.

IBM Japan Ltd.

19-21, Nihonbashi-Hakozakicho, Chuo-ku

Tokio 103-8510, Japón

El párrafo siguiente no se aplica al Reino Unido ni a ningún otro país donde las disposiciones en él expuestas sean incompatibles con la legislación local: INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL" SIN GARANTÍA DE NINGUNA CLASE, YA SEA EXPLÍCITA O IMPLÍCITA, INCLUIDAS, PERO SIN LIMITARSE A ELLAS, LAS GARANTÍAS IMPLÍCITAS DE NO VULNERABILIDAD, COMERCIALIZACIÓN O IDONEIDAD PARA UN PROPÓSITO DETERMINADO. Algunos países no permiten la renuncia a garantías explícitas o implícitas en determinadas transacciones, por lo que puede que esta declaración no sea aplicable en su caso.

La información de este documento puede incluir imprecisiones técnicas o errores tipográficos. Periódicamente se efectúan cambios en la información aquí contenida; estos cambios se incorporarán en nuevas ediciones de la publicación. IBM puede reservarse el derecho de realizar mejoras y/o cambios en los productos y/o programas descritos en esta publicación en cualquier momento sin previo aviso.

© Copyright IBM Corp. 2012, 2013

Cualquier referencia incluida en esta información a sitios web que no sean de IBM sólo se proporciona para su comodidad y en ningún modo constituye una aprobación de dichos sitios web. El material de esos sitios web no forma parte del material de este producto de IBM y la utilización de esos sitios web se realizará bajo su total responsabilidad.

IBM puede utilizar o distribuir cualquier información que se le facilite de la manera que considere adecuada, sin contraer por ello ninguna obligación con el remitente. Los titulares de licencias de este programa que deseen obtener información sobre el mismo con el fin de permitir: (i) el intercambio de información entre programas creados independientemente y otros programas (incluido éste) y el uso mutuo de información que se haya intercambiado, deben ponerse en contacto con:

IBM Corporation

Dept F6, Bldg 1

294 Route 100

Somers NY 10589-3216

EE.UU.

Dicha información puede estar disponible, sujeta a los términos y condiciones apropiados, incluido en algunos casos el pago de una cuota.

IBM proporciona el programa bajo licencia que se describe en este documento y todo el material bajo licencia disponible para el mismo bajo los términos del Acuerdo de cliente de IBM, el Acuerdo internacional de licencias de programas de IBM o cualquier acuerdo equivalente entre las partes.

Los datos de rendimiento incluidos aquí se determinaron en un entorno controlado. Por lo tanto, los resultados obtenidos en otros entornos operativos pueden variar considerablemente. Algunas mediciones podrían haberse realizado en sistemas en desarrollo y, por lo tanto, no existe ningún tipo de garantía de que dichas mediciones sean las mismas en los sistemas con disponibilidad general. Además, es posible que algunas mediciones se hayan calculado mediante extrapolación. Los resultados reales pueden variar. Los usuarios de este documento deben verificar los datos aplicables a sus entornos específicos.

La información relacionada con productos que no son de IBM se ha obtenido de los proveedores de dichos productos, de sus anuncios publicados o de otras fuentes de disponibilidad pública.

IBM no ha probado estos productos y no puede confirmar la precisión de rendimiento, compatibilidad ni otras afirmaciones relacionadas con productos que no son de IBM. Las preguntas relativas a las prestaciones de productos que no son de IBM deben dirigirse a los proveedores de dichos productos.

Las afirmaciones relativas a las intenciones futuras de IBM están sujetas a cambio o retirada sin previo aviso, y sólo representan objetivos

Todos los precios de IBM que se muestran son precios de distribuidor recomendados por IBM, corresponden al momento actual y están sujetos a cambios sin aviso previo. Los precios de los distribuidores pueden variar.

Esta información se ofrece con fines de planificación únicamente. La información incluida en este documento puede cambiar antes de que los productos descritos estén disponibles.

Esta información contiene ejemplos de datos e informes utilizados en operaciones comerciales diarias. Para ilustrarlos de la manera más completa posible, los ejemplos incluyen los nombres de personas, empresas, marcas y productos. Todos estos nombres son ficticios y cualquier parecido con nombres y direcciones utilizados por empresas comerciales reales son mera coincidencia.

LICENCIA DE COPYRIGHT:

Esta información contiene programas de aplicación de ejemplo en lenguaje fuente, que ilustran técnicas de programación en diversas plataformas operativas. Puede copiar, modificar y distribuir los programas de ejemplo de cualquier forma, sin tener que pagar a IBM, con intención de desarrollar, utilizar, comercializar o distribuir programas de aplicación que estén en conformidad con la interfaz de programación de aplicaciones (API) de la plataforma operativa para la que están escritos los programas de ejemplo. Estos ejemplos no se han probado exhaustivamente bajo todas las condiciones. Por lo tanto, IBM no puede garantizar ni implicar la fiabilidad, capacidad de servicio o función de estos programas. Los programas de ejemplo se proporcionan "TAL CUAL", sin garantía de ningún tipo. IBM no es responsable de ningún daño resultante de la utilización de los programas de ejemplo por parte del usuario.

Todas las copias o fragmentos de las copias de estos programas de ejemplo o cualquier trabajo que de ellos se derive, deberán incluir un aviso de copyright como el que se indica a continuación:

© (el nombre de la empresa) (año). Algunas partes de este código proceden de los programas de ejemplo de IBM Corp.

© Copyright IBM Corp. _escriba el año o los años_. Reservados todos los derechos.

Si visualiza esta información en una copia software, es posible que no aparezcan las fotografías ni las ilustraciones en color.

Información de la interfaz de programación

Esta publicación documenta interfaces de programación que permiten al cliente escribir programas para obtener los servicios de IBM Cúram Social Program Management.

Marcas registradas

IBM, el logotipo de IBM e ibm.com son marcas registradas de International Business Machines Corp., registradas en muchas jurisdicciones en todo el mundo. Otros nombres de productos y servicios pueden ser marcas registradas de IBM u otras empresas. Encontrará una lista actual de marcas registradas de IBM en la web en "Copyright and trademark information" en http://www.ibm.com/legal/us/en/copytrade.shtml.

Apache es una marca registrada de Apache Software Foundation.

Oracle, WebLogic Server, Java y todas las marcas registradas y logotipos basados en Java son marcas registradas de Oracle y/o sus filiales.

Otros nombres pueden ser marcas registradas de sus respectivos propietarios. Otros nombres de empresas, productos o servicios pueden ser marcas registradas o de servicio de terceros.

Impreso en España