IBM Cúram Social Program Management

Cúram Security

Version 6.0.5

IBM Cúram Social Program Management

Cúram Security

Version 6.0.5

Hinweis

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die Informationen unter „Bemerkungen" auf Seite 61 gelesen werden.

Überarbeitung: Mai 2013

Diese Ausgabe bezieht sich auf IBM Cúram Social Program Management Version 6.0.5 und alle nachfolgenden Releases, sofern nicht anderweitig in neuen Ausgaben angegeben.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellen

Kapitel 1. Cúram Security

1.1 Zweck

In diesem Handbuch werden die Sicherheitsaspekte beschrieben, die bei der Entwicklung und Bereitstellung einer IBM Cúram Social Program Management-Unternehmensanwendung berücksichtigt werden müssen. Der Begriff "Sicherheit" wird für die Beschreibung vieler unterschiedlicher Bereiche verwendet. In diesem Dokument werden die folgenden Bereiche abgedeckt: Authentifizierung und Berechtigung. Außerdem werden im vorliegenden Handbuch die sicherbaren Elemente von IBM Cúram Social Program Management-Anwendungen beschrieben.

Das vorliegende Handbuch besteht aus zwei Teilen. Der erste Teil enthält eine Übersicht der Sicherheitsbereiche und sollte sowohl von technischen Architekten als auch von Anwendungsentwicklern gelesen werden. Im zweiten Teil sind Informationen zur Vorgehensweise bei der Entwicklung sowie Beispiele für die Anwendungssicherheit enthalten.

1.2 Zielgruppe

Es gibt zwei Hauptzielgruppen, an die sich dieses Dokument richtet:

- Technische Architekten, die die Integration in andere Systeme zur Bereitstellungszeit berücksichtigen müssen, z. B. LDAP.
- Anwendungsentwickler, die den Typ der zu entwickelnden Anwendung berücksichtigen müssen, d. h.: Richtet sich die Anwendung an interne und/oder externe Benutzer?

Anmerkung: Bei internen Benutzern handelt es sich um Benutzer, die in der Cúram-Benutzerdatenbanktabelle existieren. Sie sind Teil der Organisation und sind in der Regel dafür zuständig, Forderungen für Beteiligte zu verwalten. Externe Benutzer sind alle anderen Typen von Benutzern. Externe Benutzer sind nicht Teil der Organisation. Ihr Zugang ist eingeschränkt. Ein Beispiel für einen externen Benutzer wäre ein Anbieter, der der Organisation einen Service bereitstellt.

1.3 Übersicht

Die Sicherheit ist in der Infrastruktur integriert, auf der die Entwicklung der IBM Cúram Social Program Management-Anwendung basiert. Sie unterstützt die Authentifizierung eines Benutzers bei der Anmeldung und stellt Unterstützung für den Berechtigungsprozess bereit. Der Entwurf einer Anwendung ist erst vollständig, wenn als Erstes die Auswirkungen eines Schutzes der Anwendung gegen unbefugten Zugriff auf vertrauliche Daten oder die Funktion berücksichtigt werden. Die Sicherheit ist somit einer der zentralen Schwerpunkte bei der Anwendungsentwicklung.

Im Folgenden sind die grundlegenden Konzepte der IBM Cúram Social Program Management-Sicherheit aufgeführt:

- Authentifizierung
- Berechtigung

Darüber hinaus gibt es das Konzept der standortbasierten Sicherheit. Auf Organisationsebene schränkt die Standortsicherheit den Benutzerzugriff auf Kunden- und Fallinformationen ein. Außerdem kann die Datensicherheit des Standorts konfiguriert werden, um Benutzern den Zugriff auf andere Standorte als den eigenen zu ermöglichen. Im vorliegenden Handbuch wird die standortbasierte Sicherheit nicht ausführlich beschrieben. Ziehen Sie das Handbuch *Cúram Location Administration* zurate, wenn Sie weitere Details zur standortbasierten Sicherheit wünschen.

1.4 Kapitel in diesem Handbuch

Das vorliegende Handbuch besteht aus zwei Teilen. Teil 1 besteht aus den folgenden Kapiteln, die eine allgemeine Übersicht über die Sicherheitsarchitektur von IBM Cúram Social Program Management für die Anwendung und Bereitstellung für Anwendungsserver bieten:

Kapitel 2 - Authentifizierung
In diesem Kapitel wird die Authentifizierungsarchitektur für IBM Cúram Social Program Management beschrieben. hier werden alle Bereiche und sämtliche verfügbaren, anpassbaren Elemente ausführlich beschrieben.

Kapitel 3 - Berechtigung
In diesem Kapitel wird die Funktionsweise der Berechtigung und deren Konfiguration für Benutzer beschrieben.

Kapitel 4 - Verschlüsselung in Cúram
In diesem Kapitel wird beschrieben, wie in IBM Cúram Social Program Management eine Verschlüsselung zum Sichern von Kennwörtern verwendet wird.

Kapitel 5 - Caching von Sicherheitsdaten
In diesem Kapitel werden der integrierte Cúram-Sicherheitscache und der IBM® WebSphere Application Server-Cache sowie deren Einfluss auf die Benutzerauthentifizierung beschrieben.

Kapitel 6 - Sicherheit für alternative Clients
In diesem Kapitel werden die Benutzernamen beschrieben, die für die Cúram-Benutzerdatenbanktabelle existieren müssen, um zu gewährleisten, dass Prozesse wie Workflow erfolgreich ausgeführt werden können.

Kapitel 7 - Anwendungen für externe Benutzer
In diesem Kapitel wird beschrieben, weshalb eine Anwendung für externe Benutzer notwendig sein könnte und welche Punkte hierfür berücksichtigt werden sollten.

Kapitel 8 - Verwenden von Single Sign-on
In diesem Kapitel werden die Eigenschaften des Anwendungsservers beschrieben, die bei der Verwendung von IBM Cúram Social Program Management in einer Single Sign-on-Lösung berücksichtigt werden müssen.

Kapitel 9 - Weitere Sicherheitsaspekte
Dieses Kapitel enthält eine kurze Übersicht zu einigen externen Sicherheitsverfahren und -aspekten.

Teil 2 besteht aus den folgenden Kapiteln, in denen verschiedene Vorgehensweisen im Hinblick auf Entwicklungsaktivitäten für die Codierung und Anpassung der Sicherheit beschrieben werden:

Kapitel 10 - Anpassen der Authentifizierung
In diesem Kapitel werden Anpassungspunkte und Entwicklungsartefakte beschrieben, die für die Authentifizierung relevant sind.

Kapitel 11 - Anpassen der Berechtigung
In diesem Kapitel wird die Vorgehensweise bei der Implementierung der Berechtigung für IBM Cúram Social Program Management beschrieben.

Kapitel 12 - Anpassen der Verschlüsselung
In diesem Kapitel wird beschrieben, wie die Cúram-Verschlüsselung angepasst wird.

Kapitel 13 - Anpassen von Anwendungen für externe Benutzer
In diesem Kapitel wird beschrieben, wie eine Anwendung für externe Benutzer entwickelt wird.

Kapitel 2. Authentifizierung

2.1 Übersicht

In diesem Kapitel wird die Authentifizierung für IBM Cúram Social Program Management beschrieben. Authentifizierung ist der Prozess, mit dem die Identität von Benutzern überprüft wird. Sie ist immer dann erforderlich, wenn die Identität eines Benutzers verifiziert werden muss, damit dieser auf eine gesicherte Ressource in einem System zugreifen kann.

Bei der formularbasierten Authentifizierung wird einem Benutzer ein Formular angezeigt, in das er die Berechtigungsnachweise für einen Benutzernamen und ein Kennwort eingeben kann. Diese Berechtigungsnachweise werden mit den auf dem System gespeicherten Berechtigungsnachweisen für diesen Benutzernamen verglichen; wenn sie übereinstimmen, wird der Benutzer als authentifizierter Benutzer für das System angesehen. Aus Sicherheitsgründen wird das Kennwort für die Authentifizierung eines Benutzers auf dem System in mittels Digest verschlüsselter Form gespeichert.

Der Web-Client von IBM Cúram Social Program Management ist so konfiguriert, dass er die formularbasierte Authentifizierung unterstützt; dies bedeutet, dass bevor ein Benutzer auf den Inhalt des Web-Clients zugreifen kann, er für die Authentifizierung an ein Anmeldeformular weitergeleitet wird.

Der Authentifizierungsprozess umfasst die Überprüfung des Benutzernamens und des Kennworts; er wird standardmäßig von einem JAAS-Anmeldemodul durchgeführt (JAAS = Java™ Authentication and Authorization Service). HTTPS/SSL ist auf dem Web-Client standardmäßig aktiviert, um zu gewährleisten, dass der Authentifizierungsmodus zur formularbasierten Anmeldung gesichert ist.

2.2 Authentifizierung

Mithilfe des JAAS-Anmeldemoduls von Cúram können unterschiedliche Authentifizierungsmodi konfiguriert werden (in Abhängigkeit von den Authentifizierungsanforderungen).

Die folgenden Authentifizierungsmodi werden unterstützt:

- Standardauthentifizierung
- Ausschließliche Authentifizierung nach Identität
- Externer Zugriff - Sicherheitsauthentifizierung

Diese Modi werden in den nachfolgenden Abschnitten ausführlich beschrieben.

2.3 Authentifizierungsarchitektur

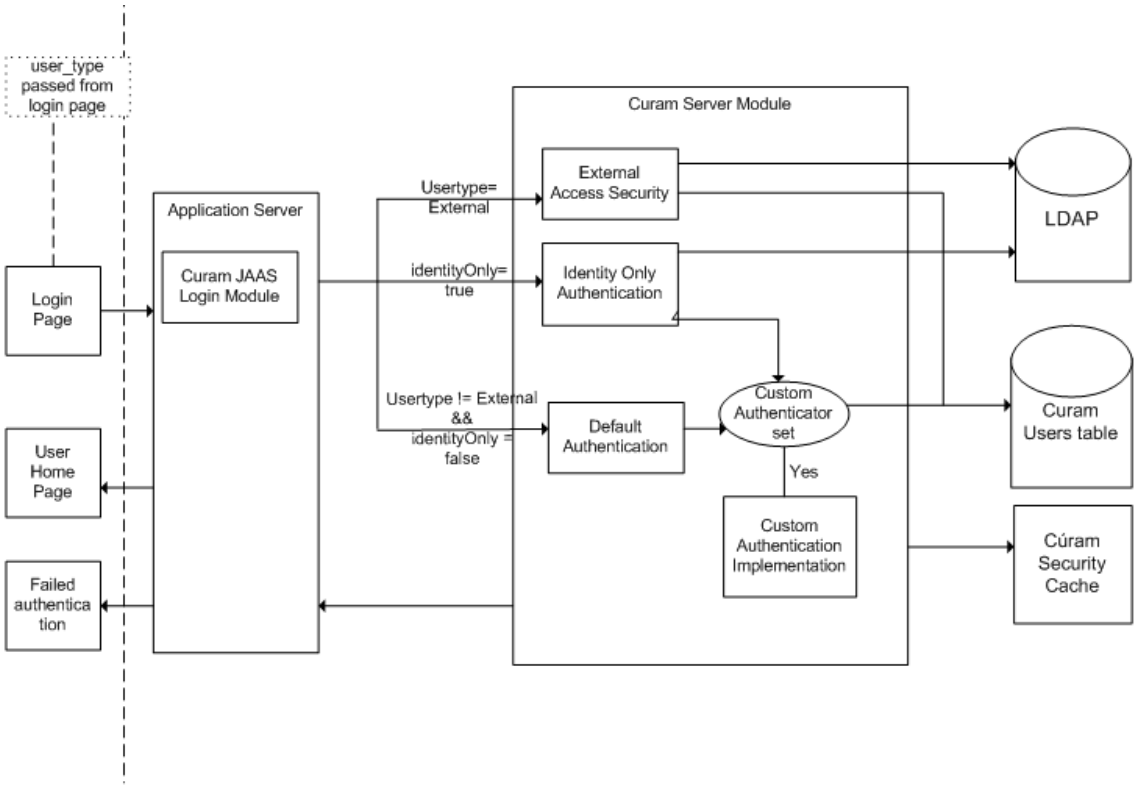

Abbildung 1. Authentifizierungsarchitektur

Die oben beschriebene 2.3, „Authentifizierungsarchitektur" ist ein Entwurf für die Architektur des Authentifizierungsprozesses für einen Benutzer. Standardmäßig wird für einen Benutzer die Standardauthentifizierung durchgeführt. Dieses Verhalten kann in Abhängigkeit von den Authentifizierungsanforderungen sowohl für interne als auch für externe Benutzer angepasst werden. In den folgenden Abschnitten dieses Kapitels werden alle Funktionsbereiche ausführlich beschrieben, die die Authentifizierungsarchitektur bilden, und es wird angegeben, an welchen Stellen Anpassungen möglich sind.

2.4 Standardauthentifizierung

Die Standardauthentifizierung für IBM Cúram Social Program Management umfasst die Benutzeranmeldung über die Anmeldeanzeige, in der der Benutzer zur Eingabe der Berechtigungsnachweise in Form von Benutzername und Kennwort aufgefordert wird. Diese Berechtigungsnachweise werden dann an das JAAS-Anmeldemodul von Cúram übergeben, das auf dem Anwendungsserver konfiguriert wurde.

Die Standardauthentifizierung wird aufgerufen und der eingegebene Benutzername und das eingegebene Kennwort werden mit dem in der Cúram-Benutzerdatenbanktabelle gespeicherten Benutzernamen und Kennwort abgeglichen. Der Cúram-Benutzername ist nicht veränderbar, aber Sie haben die Möglichkeit, Ihr System so zu konfigurieren, dass eine Cúram-Anmelde-ID verwendet wird, die geändert werden kann. Die Anmelde-ID ist eine logische Erweiterung des Cúram-Benutzers. Es werden sowohl für den Benutzernamen als auch für die Anmelde-ID dieselben Überprüfungen durchgeführt. Weitere Informationen zu alternativen Anmelde-IDs enthält 2.5, „Alternative Anmelde-IDs", auf Seite 5.

Bei der Authentifizierung werden eine Reihe von Überprüfungen der Anmeldeinformationen durchgeführt. Weitere Informationen zu diesen Überprüfungen enthält 2.14, „Standardauthentifizierung", auf Seite 11.

Angenommen, es werden alle Überprüfungen erfolgreich abgeschlossen, dann wird der Benutzer von der Anwendung als authentifiziert betrachtet.

Sobald der Benutzer authentifiziert ist, wird er zum Cúram-Sicherheitscache hinzugefügt. Im Cúram-Sicherheitscache werden der Benutzername und alle zugehörigen Berechtigungsdaten für diesen Benutzer gespeichert, um die Abfrage dieser Daten für einen Benutzer zu optimieren. In Kapitel 5, „Caching von Sicherheitsdaten", auf Seite 23 finden Sie weitere Details zum Cúram-Sicherheitscache. In Abbildung 2.3 unten ist der Pfad für die Standardauthentifizierung hervorgehoben.

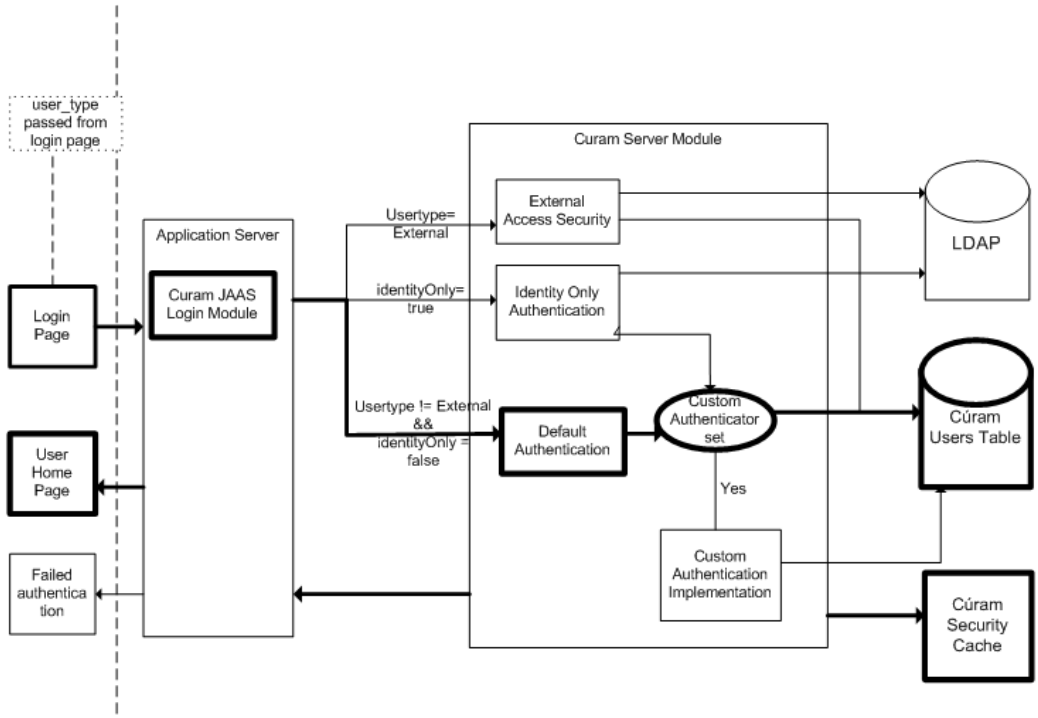

Abbildung 2. Standardauthentifizierung

2.5 Alternative Anmelde-IDs

In Cúram werden standardmäßig der Benutzername und das mittels Digest verschlüsselte Kennwort in der Tabelle "Users" für die Authentifizierung verwendet. Dieser Benutzername ist nicht veränderbar, d. h., sobald er erstellt wurde, kann er nicht mehr geändert werden. Diese fehlende Flexibilität erfüllt möglicherweise nicht die Anforderungen einiger Installationen. Sie können jedoch statt eines nicht veränderbaren Benutzernamens eine Anmelde-ID verwenden, die aktualisiert werden kann.Die Anmelde-ID fungiert als logische Erweiterung der Cúram-Tabelle "Users". Wenn die alternative Anmelde-ID verwendet wird, bleibt der Benutzername weiterhin bestehen und wird von Cúram intern verwendet, der Benutzer meldet sich jedoch mit der Anmelde-ID bei Cúram an.

Hinweise zur Verwendung der alternativen Anmelde-ID:

- Es kann nur eine alternative Anmelde-ID oder ein Benutzername (aber nicht beides) verwendet werden. Dies bedeutet, es kann keine Cúram-Benutzer mit Benutzernamen und Cúram-Benutzer mit Anmelde-IDs nebeneinander geben.
- Die Cúram-Tabelle "ExtendedUsersInfo", in der die Anmelde-ID gespeichert ist, muss gefüllt werden, bevor die Funktion für alternative Anmelde-IDs aktiviert wird. Dies wird nachstehend genauer erläutert.
- Bei der Verwendung einer Authentifizierung per Anmelde-ID werden die Ergebnisse in der Tabelle "AuthenticationLog" gespeichert. In der Spalte "AltLogin" wird angegeben, ob in der Spalte "UserName" ein Benutzername (false) oder eine Anmelde-ID (true) eingetragen ist.

- Anmelde-IDs gelten nur für interne Cúram-Benutzer, d. h. Benutzer, die in der Cúram-Tabelle "Users" gespeichert sind. Wenn Sie jedoch eine ausschließliche Authentifizierung nach Identität mithilfe von Anmelde-IDs durchführen, müssen diese IDs unabhängig vom Speicherort (z. B. WebSphere-Registry, LDAP usw.) den Anmelde-IDs entsprechen, die in der Cúram-Tabelle "ExtendedUsersInfo" gespeichert sind.
- Bei der Zuweisung von Anmelde-IDs müssen Sie auf IDs achten, die intern verwendet werden bzw. die Abhängigkeiten (z. B. von Eigenschaftswerten) außerhalb der Cúram-Tabelle "Users" haben. Folgende Benutzernamen können Probleme verursachen, wenn ihre Anmelde-IDs vom Benutzernamen ohne entsprechende Änderung abweichen:
  - SYSTEM - In WebSphere gehört dieser Benutzername zur JMS-Verarbeitung und ist Teil der WebSphere-Konfiguration zum Zeitpunkt der Anwendungsimplementierung. Informationen zum Ändern dieser ID finden Sie in 6.2, „Obligatorische Cúram-Benutzer", auf Seite 25 sowie im entsprechenden Handbuch WebSphere *Cúram Deployment*.
  - DBTOJMS - Dies ist der Standard-DBtoJMS-Benutzername, der von der Stapelverarbeitung verwendet und von der Eigenschaft `curam.security.credentials.dbtojms.username` referenziert wird. Weitere Informationen finden Sie in 6.2, „Obligatorische Cúram-Benutzer", auf Seite 25, 6.5, „JMS-Messaging", auf Seite 26, 6.6, „Verzögerte Verarbeitung", auf Seite 26 sowie im Handbuch *Cúram Batch Processing*.
  - WEBSVCS - Dies ist der Standard-Web-Services-Benutzername, der von der Eigenschaft `curam.security.credentials.dbtojms.username` referenziert wird. Weitere Informationen finden Sie in 6.2, „Obligatorische Cúram-Benutzer", auf Seite 25, 6.3, „Web-Services", auf Seite 25 sowie im Handbuch *Cúram Web Services*.
  - unauthenticated - Dies ist der WebSphere-Hauptbenutzername für nicht authentifizierte Benutzer. Diese Anmelde-ID darf nicht geändert werden.

Um die Verwendung alternativer Anmelde-IDs zu aktivieren, sobald die Tabelle "ExtendedUsersInfo" gefüllt wurde, legen Sie die Eigenschaft `curam.security.altlogin.enabled` auf `true` fest. (Weitere Informationen zu den Cúram-Eigenschaften finden Sie im Handbuch *Cúram Server Developer*.) Hier handelt es sich um eine statische Eigenschaft, daher muss Cúram neu gestartet werden, um die Eigenschaft zu übernehmen.

Ihnen stehen zum Füllen der Tabelle "ExtendedUsersInfo", bevor Sie die Funktion aktivieren, einige Optionen zur Verfügung. Beispielsweise:

- Mit einer einfachen SQL-Anweisung können Sie die Tabelle mit dem Benutzernamen in der Tabelle "Users" füllen. Daher wirkt sich dies nicht unmittelbar auf den Benutzer aus: `INSERT INTO EXTENDEDUSERSINFO (USERNAME, LOGINID, UPPERLOGINID, VERSIONNO) (SELECT USERNAME, USERNAME, UPPER(USERNAME), 1 FROM USERS);` Anschließend können Sie Ihre Modifikationen an den Anmelde-IDs kontrolliert implementieren.
- Sie können eine SQL-Anwendung implementieren, die Ihre Benutzernamen- und Anmelde-ID-Zuordnung (z. B. allgemeine LDAP-Namen) implementieren.

  Anmerkung: Sie müssen die Fremdschlüssel-Beziehung des Benutzernamens zwischen den Tabellen "Users" und "ExtendedUsersInfo" beibehalten.

2.6 Anmeldeseite

Die Standardanmeldeseite wird durch die Datei `logon.jsp` dargestellt. Die Datei `logon.jsp` stellt die Anmeldeseite für den Benutzer dar, auf der dieser die formularbasierte Anmeldungsauthentifizierung ausführen kann. Standardmäßig enthält die Datei `logon.jsp` die Felder für den Benutzernamen und das Kennwort. Die Datei `logon.jsp` kann allerdings so angepasst werden, dass ein weiterer Parameter eingegeben werden kann; dabei wird das Feld "user_type" hinzugefügt. Durch dieses Feld wird der Typ des Benutzers festgelegt, der sich anmeldet, d. h. interner oder externer Benutzer. Benutzername, Kennwort und Benutzertyp (falls vorhanden) werden im Rahmen des Authentifizierungsprozesses an das JAAS-Anmeldemodul von Cúram übergeben.

In der Standarddatei `logon.jsp` ist die Eigenschaft "user_type" nicht gesetzt. Wird diese Eigenschaft nicht verwendet, wird angenommen, dass der Benutzer ein interner Benutzer ist. Wenn diese Eigenschaft gesetzt ist, bedeutet dies, dass sich ein externer Benutzer anmeldet. Diese Eigenschaft kann auf einen beliebigen Wert gesetzt werden, mit Ausnahme von "INTERNAL".

2.7 Anpassung der Anmeldeseite

Die Datei `logon.jsp` kann angepasst werden, d. h., sie kann aus zahlreichen Gründen vollständig durch eine angepasste Datei des Typs `logon.jsp` ersetzt werden:

Es wird eine Clientanwendung für einen externen Benutzer entwickelt.
Wenn eine Clientanwendung für einen externen Benutzer entwickelt wird, muss eine neue Datei des Typs `logon.jsp` erstellt werden, da der Benutzertyp so definiert werden muss, dass er angibt, dass sich ein externer Benutzer anmeldet. In 13.3, „Erstellen einer Anmeldeseite für externe Benutzerclients", auf Seite 51 finden Sie weitere Details.

Es ist eine automatische Anmeldung erforderlich.
Für einige Clientanwendungen für externe Benutzer ist keine Benutzerauthentifizierung erforderlich; folglich sollten ein Benutzername und ein Kennwort nicht angefordert werden, d. h. im Fall einer Anwendung mit externem öffentlichen Zugriff. Es ist nicht möglich, die Authentifizierung zu inaktivieren; für diese Voraussetzung ist es daher das Einfachste, ein automatisches Anmeldescript zu schreiben. Dies wird durch Anpassen der Datei `logon.jsp` für die Anwendung mit externem öffentlichen Zugriff erreicht. In 13.4, „Erstellen einer automatischen Anmeldeseite für externe Benutzerclients", auf Seite 51 finden Sie weitere Details.

Es ist eine andere Darstellung erforderlich.
Im Abschnitt zu den Anmeldeseiten im Referenzhandbuch zu *Cúram Web Client* finden Sie weitere Details zur Darstellung der Datei `logon.jsp`.

Für Benutzernamen besteht die Voraussetzung, dass sie Erweiterungszeichen enthalten müssen (nur bei Oracle WebLogic Server).
WebLogic Server stellt das proprietäre Attribut **j_character_encoding** bereit, das zur Datei `logon.jsp` hinzugefügt werden muss. In 10.3, „Aktivieren von Benutzernamen mit Erweiterungszeichen für WebLogic Server", auf Seite 37 finden Sie weitere Details.

2.8 Cúram-JAAS-Anmeldemodul

Die Authentifizierung erfolgt durch ein JAAS-Anmeldemodul. Es wird auf dem Anwendungsserver konfiguriert und im Rahmen des Authentifizierungsprozesses automatisch vom Anwendungsserver für beliebigen Zugriff auf die IBM Cúram Social Program Management-Anwendung aufgerufen. Der Vorteil dieses Ansatzes ist, dass der Standardauthentifizierungsmechanismus mit einer angepassten Methode verwendet oder durch eine solche ersetzt werden kann, ohne Auswirkungen auf die IBM Cúram Social Program Management-Anwendung zu haben.

Wie bereits erwähnt, kann das Cúram-JAAS-Anmeldemodul so konfiguriert werden, dass es in drei Modi ausgeführt wird. Weitere Informationen zur Konfiguration des Anmeldemoduls und anwendungsserverspezifischem Verhalten finden Sie im Abschnitt zur Konfiguration des Anwendungsservers im Handbuch *Cúram Server Deployment* für den jeweils verwendeten Anwendungsserver.

Projektspezifische Anforderungen können bedeuten, dass mehrere Anmeldemodule erforderlich sind; so muss z. B. ein Benutzer zu Überprüfungszwecken möglicherweise mehr als nur den Benutzernamen und das Kennwort eingeben. Es ist möglich, mehrere Anmeldemodule auf dem Anwendungsserver zu konfigurieren. Jedes Anmeldemodul wird in der Reihenfolge ausgeführt, die durch die Einstellungen auf dem Anwendungsserver vorgegeben ist. Weitere Informationen zu diesen Einstellungen finden Sie in der Dokumentation zu WebSphere oder WebLogic Server.

Sobald der Benutzer erfolgreich durch alle Anmeldemodule authentifiziert wurde, für die eine erfolgreiche Authentifizierung des Benutzers erforderlich ist (konfigurierbar auf dem Anwendungsserver), wird der Benutzer von der Anwendung als authentifiziert betrachtet.

2.9 Kennwortmanagement

Die Kennwörter aller interner und externer Cúram-Benutzer werden im Digest-Format in den Cúram-Datenbanktabellen "Users" und "ExternalUsers" gespeichert. Wenn das Cúram-JAAS-Anmeldemodul das Kennwort empfängt, wird das Kennwort mittels Digest verschlüsselt und dann an die Anmelde-Bean gesendet. Diese Verschlüsselung mittels Digest ist eine Ein-Weg-Verschlüsselung, um die Sicherheit des Kennworts sicherzustellen. Das für den Benutzer in der Datenbank gespeicherte Kennwort wird mithilfe desselben Digest-Algorithmus verschlüsselt, der Ihren Verschlüsselungseigenschaften unterliegt, wodurch sichergestellt wird, dass die verschlüsselten Kennwörter erfolgreich miteinander verglichen werden können, aber sicher bleiben.

Benutzer, die extern verwaltet werden, z. B. über LDAP mit einer Konfiguration für die ausschließliche Authentifizierung nach Identität, unterliegen dem oben genannten Prozess nicht. Bei der Authentifizierung gegen ein Drittanbietersystem (z. B. LDAP oder ein SSO-Server), bei der die Cúram-Anwendung die vom Benutzer eingegebenen Berechtigungsnachweise für das Drittanbietersystem übergeben muss, kann die angepasste Implementierung von `curam.util.security.PublicAccessUser` verwendet werden, da auf die Berechtigungsnachweise mit Klartext-Kennwort zugegriffen werden kann.

2.10 Standardkonfiguration für WebLogic Server

Das Cúram-JAAS-Anmeldemodul wird als Authentifizierungsprovider in WebLogic Server konfiguriert. Bei dem Cúram-Authentifizierungsprovider handelt es sich um den einzigen Provider, der durch die für WebLogic Server bereitgestellten Konfigurationsscripts konfiguriert wird. Da dies der einzige konfigurierte Authentifizierungsprovider ist, ist der Cúram-Authentifizierungsprovider für die Authentifizierung und Überprüfung des Benutzers zuständig. Wie bereits erwähnt, kann es sein, dass mehr als ein Authentifizierungsprovider in WebLogic Server konfiguriert ist; in diesem Fall ist der Cúram-Authentifizierungsprovider möglicherweise nicht für die Authentifizierung und Überprüfung des Benutzers zuständig. In 8.3, „Single Sign-on für WebLogic Server", auf Seite 34 finden Sie weitere Details.

2.11 Standardkonfiguration für WebSphere

Das Cúram-JAAS-Anmeldemodul wird als Systemanmeldemodul in WebSphere konfiguriert. Die standardmäßige scriptgesteuerte Sicherheitskonfiguration in WebSphere umfasst die standardmäßige dateibasierte Benutzerregistry und das Cúram-Systemanmeldemodul. Bei der Benutzerregistry in WebSphere handelt es sich um den Standardauthentifizierungsmechanismus; sie kann wie folgt konfiguriert werden:

- Als angepasste Benutzerregistry
- Als LDAP-Verzeichnisserver
- Als lokales Betriebssystem
- Als dateibasiertes WebSphere-Repository

Es gibt mehrere Systemanmeldekonfigurationen für WebSphere. Das Cúram-Systemanmeldemodul ist für die Konfigurationen des Typs `DEFAULT`, `WEB_INBOUND` und `RMI_INBOUND` konfiguriert. Für alle drei Konfigurationen wird ein und dasselbe Anmeldemodul verwendet. WebSphere ruft automatisch die Anmeldemodule auf, die unter bestimmten Umständen als Systemanmeldemodule konfiguriert werden:

- `DEFAULT`

  Die für die Konfiguration des Typs DEFAULT angegebenen Anmeldemodule werden für die Authentifizierung von Web-Services und JMS-Aufrufen aufgerufen. Sie werden auch während der Startphase von WebSphere aufgerufen.
- `WEB_INBOUND`

Die für die Konfiguration des Typs WEB_INBOUND angegebenen Anmeldemodule werden für die Authentifizierung von Webanforderungen verwendet.

- `RMI_INBOUND`

  Die für die Konfiguration des Typs RMI_INBOUND angegebenen Anmeldemodule werden für die Authentifizierung von Java-Clients verwendet.

Das JAAS-Anmeldemodul von Cúram existiert als Anmeldemodul innerhalb einer Kette logischer Module, die in WebSphere eingerichtet wurden. Mindestens eines dieser Anmeldemodule sollte für das Hinzufügen von Berechtigungsnachweisen für den Benutzer zuständig sein. Das Cúram-Anmeldemodul fügt standardmäßig Berechtigungsnachweise für einen authentifizierten Benutzer hinzu. Daraus ergibt sich, dass die konfigurierte WebSphere-Benutzerregistry, die von einem nachfolgenden Anmeldemodul ausgeführt wird, keine Berechtigungsnachweise hinzufügt. Deshalb ist es nicht erforderlich, Cúram-Benutzer in der WebSphere-Benutzerregistry zu definieren. Dieses Verhalten kann mithilfe der in der Datei `AppServer.properties` gesetzten Eigenschaft "curam.security.user.registry.enabled" konfiguriert werden. Weitere Details zum Einstellen dieser Eigenschaft finden Sie im Handbuch *Cúram Deployment for WebSphere Application Server* bzw. im Handbuch *Cúram Deployment for WebSphere Application Server on z/OS*. In 2.11, „Standardkonfiguration für WebSphere", auf Seite 8 unten wird der Standardauthentifizierungsablauf für WebSphere dargestellt. In 2.11, „Standardkonfiguration für WebSphere", auf Seite 8 unten wird der Authentifizierungsablauf für WebSphere dargestellt, bei dem auch die Benutzerregistry abgefragt wird, d. h., wo die Eigenschaft "curam.security.user.registry.enabled" auf `true` gesetzt ist.

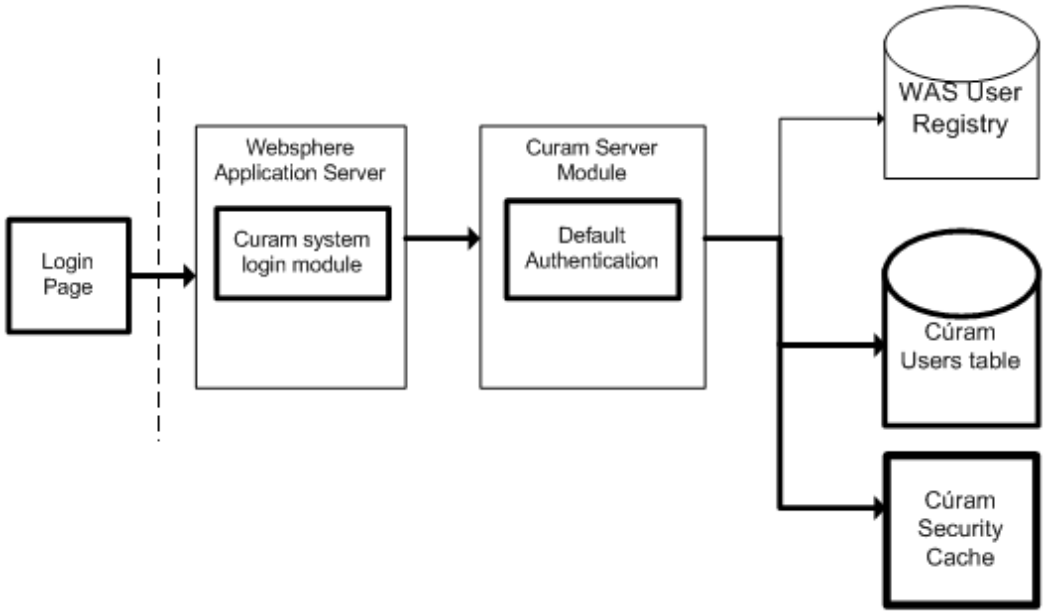

Abbildung 3. Standardauthentifizierungsablauf für WebSphere

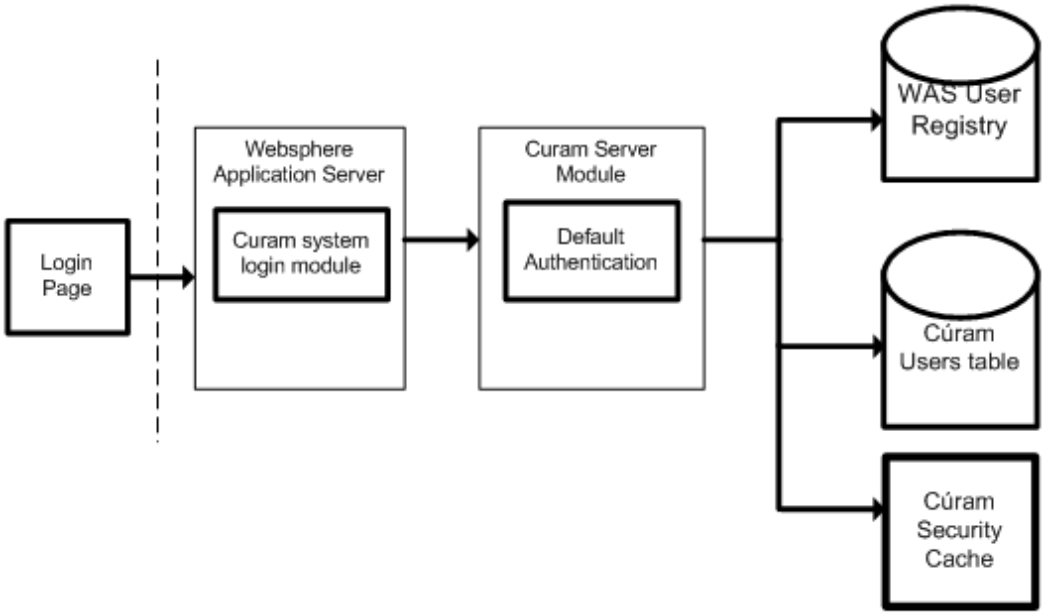

Abbildung 4. Authentifizierungsablauf für WebSphere mit aktivierter Benutzerregistry

Im Rahmen der Sicherheitskonfiguration gibt es bestimmte Benutzer, die von der Authentifizierung ausgeschlossen sind; für diese Benutzer *wird* die konfigurierte Benutzerregistry jedoch abgefragt. Die Liste der Benutzer wird automatisch als WebSphere-Sicherheitsbenutzer konfiguriert, was durch die Eigenschaft "security.username" in der Datei `AppServer.properties` angegeben wird, sowie als Datenbankbenutzer, was durch die Eigenschaft "curam.db.username" in der Datei `Bootstrap.properties` angegeben wird. Bei diesen beiden Benutzern handelt es sich um klassifizierte Benutzer mit Verwaltungsaufgaben und nicht um Anwendungsbenutzer. Es ist möglich, diese Liste der ausgeschlossenen Benutzer manuell zu erweitern; weitere Informationen hierzu finden Sie im Handbuch *Cúram Deployment for WebSphere Application Server* und im Handbuch *Cúram Deployment for WebSphere Application Server on z/OS*.

Warnung: Die Benutzer "security.username" und "curam.db.username" werden mithilfe der bereitgestellten Konfigurationsscripts automatisch zum dateibasierten WebSphere-Benutzerrepository hinzugefügt. Wenn die konfigurierte WebSphere-Benutzerregistry nicht die Standardbenutzerregistry ist, müssen sich diese Benutzer in der alternativen WebSphere-Benutzerregistry befinden.

2.12 Anpassung des JAAS-Anmeldemoduls

Es ist möglich, dass das JAAS-Anmeldemodul von Cúram die Authentifizierungsanforderungen für eine bestimmte angepasste Lösung nicht unterstützt. Es ist dringend zu empfehlen, dass das JAAS-Anmeldemodul von Cúram bei der Entwicklung eines angepassten Anmeldemoduls unangetastet bleibt und mit aktivierter ausschließlicher Authentifizierung über die Identität verwendet wird. Sollte es jedoch unumgänglich sein, kann das JAAS-Anmeldemodul von Cúram entfernt und durch eine angepasste Lösung ersetzt werden. Sollte dies der Fall sein, wenden Sie sich an den Support.

Warnung: Zwar ist es möglich, das JAAS-Anmeldemodul von Cúram vollständig zu entfernen, aber Benutzer müssen aus Berechtigungsgründen in der Cúram-Benutzerdatenbanktabelle vorhanden bleiben.

Das JAAS-Anmeldemodul von Cúram fügt dem Cúram-Sicherheitscache automatisch neue Benutzer hinzu; wenn dieses Anmeldemodul jedoch durch ein angepasstes JAAS-Anmeldemodul ersetzt wird, wird diese Funktionalität entfernt. Wenn ein angepasstes JAAS-Anmeldemodul das JAAS-Anmeldemodul von Cúram vollständig ersetzt, ist das angepasste JAAS-Anmeldemodul dafür zuständig, zu gewährleisten, dass eine Aktualisierung des Sicherheitscache ausgelöst wird, sobald ein neuer Benutzer zur Datenbank hinzugefügt wird.

2.13 Überprüfungsprozess für Authentifizierung

Der Typ der durchgeführten Überprüfungen ist von dem verwendeten Authentifizierungsmodus abhängig. Unten sehen Sie eine Liste der Authentifizierungsmodi/-konfigurationen und vollständige Details zu den für die einzelnen Authentifizierungsmodi durchgeführten Überprüfungen.

2.14 Standardauthentifizierung

Die Standardauthentifizierung ist Bestandteil der Standardkonfiguration; dieser Authentifizierungsmodus umfasst die Überprüfung des Benutzernamens und des Kennworts, die bei der Anmeldung für die Cúram-Benutzerdatenbanktabelle angegeben wurden. Alle Anmeldeinformationen werden in diesem Fall von der IBM Cúram Social Program Management-Anwendung gepflegt.

2.15 Standardüberprüfungsprozess

Während der Standardauthentifizierung werden mit dem Cúram-Anmeldemodul folgende Elemente überprüft:

- Benutzername und Kennwort
- Ablaufdatum von Konto und/oder Kennwort
- Synchronisation des Benutzernamens mit dem Sicherheitscache
- Erkennung von Angriffen von außen, z. B. Obergrenze für die wiederholte Eingabe von Kennwörtern, falsche Benutzernamen, Fehler bei Kennwortänderungen
- Tag und Uhrzeit von Zugriffsbeschränkungen - Tag der Woche und Zeitraum an diesem Tag

Bei der Authentifizierung und Berechtigung von Benutzernamen ist standardmäßig die Groß-/Kleinschreibung zu beachten; es ist jedoch möglich, die Beachtung der Groß-/Kleinschreibung für die Authentifizierung zu inaktivieren. Wenn doppelte Benutzernamen ohne Beachtung der Groß-/Kleinschreibung existieren (z. B. caseworker, CaseWorker), schlägt die Authentifizierung aufgrund des mehrdeutigen Benutzernamens fehl. In 10.4, „Ändern der Groß-/Kleinschreibung des Benutzernamens”, auf Seite 37 finden Sie weitere Details hierzu.

2.16 Authentifizierungsversuche

Authentifizierungsfehler werden nicht direkt an einen Client gemeldet, da dadurch zusätzliche Informationen an einen Angreifer von außen geliefert werden würden, der versucht, in ein System einzudringen. So würde beispielsweise das Melden eines falschen Kennworts darauf hinweisen, dass der Benutzername gültig ist. Alle Authentifizierungsversuche (sowohl bei Erfolg als auch bei Fehler) werden in einer Datenbanktabelle mit dem Namen "AuthenticationLog" gespeichert. In 11.15, „Analysieren der Datenbanktabelle "AuthorisationLog"”, auf Seite 44 finden Sie weitere Details.

2.17 Anpassung der Standardauthentifizierung

Die Standardimplementierung kann so angepasst werden, dass eine veränderbare Anmelde-ID statt des Cúram-Benutzernamens verwendet wird. Es ist außerdem möglich, weitere Überprüfungen durch Implementieren des angepassten Authentifikators hinzuzufügen. (Weitere Informationen finden Sie in 2.21, „Angepasste Überprüfungen”, auf Seite 13.)

2.18 Ausschließliche Authentifizierung nach Identität

Anstelle der Standardüberprüfungen, die in 2.15, „Standardüberprüfungsprozess”, auf Seite 11 oben aufgeführt werden, kann die Authentifizierung so konfiguriert werden, dass eine ausschließliche Überprüfung nach Identität durchgeführt wird.

"Ausschließliche Überprüfung nach Identität" bedeutet, dass durch den Authentifizierungsmechanismus lediglich sichergestellt wird, dass der Benutzername für die Benutzeranmeldung in der Cúram-Benutzerdatenbanktabelle vorhanden ist. Eine vollständige Authentifizierung muss durch einen alternativen Mechanismus durchgeführt werden, damit eine Konfiguration auf dem Anwendungsserver möglich ist.

Ein LDAP-Verzeichnisserver ist ein Beispiel für einen alternativen Mechanismus; dieser wird sowohl von WebSphere Application Server als auch von WebLogic Server Application Server als Authentifizierungsmechanismus unterstützt. Eine weitere Alternative ist die Verwendung einer Single Sign-on-Lösung für die Authentifizierung oder die Implementierung eines angepassten Anmeldemoduls. Für die Arbeit mit Lösungen für angepasste Anwendungsserver sollten Sie die IBM Dokumentation oder die Oracle-Dokumentation zurate ziehen.

Bei der ausschließlichen Authentifizierung nach Identität (wie auch bei der Standardauthentifizierung) werden Einträge am Schluss des Authentifizierungsprozesses zur Datenbanktabelle "AuthenticationLog" hinzugefügt.

Bei einer erfolgreichen Anmeldung wird der folgende Status verwendet:

- `AUTHONLY`

Bei einem fehlerhaften Szenario wird der folgende Status verwendet:

- `BADUSER`

  Dies ist das einzige Szenario für einen möglichen Fehler; dabei existiert kein Benutzer.

Die Felder `loginFailures` und `lastLogin` der Datenbanktabelle "AuthenticationLog" sind nicht definiert. Dies trifft auch dann zu, wenn angepasste Überprüfungen implementiert sind.

Wenn die Angaben zum Kennwortablauf für einen Benutzer gesetzt sind (in der Cúram-Benutzerdatenbanktabelle), wird eine Warnung zum Kennwortablauf angezeigt, wenn der Ablauf kurz bevorsteht. Bei der ausschließlichen Authentifizierung nach Identität ist diese Warnung irreführend. Es ist empfehlenswert, sämtliche Felder, die sich auf die Authentifizierungsüberprüfungen beziehen (z. B. aktivierter Kennwortablauf und aktiviertes Konto), dann nicht zu verwenden, wenn die ausschließliche Authentifizierung nach Identität aktiviert ist.

Ist die ausschließliche Authentifizierung nach Identität aktiviert, wird für die Authentifizierung keine Sicherheit angewendet, jedoch für Berechtigungszwecke. Infolgedessen müssen alle Benutzer, für die Zugriff auf die Anwendung erforderlich ist, weiterhin in der Cúram-Benutzerdatenbanktabelle und im alternativen Authentifizierungsmechanismus, z. B. LDAP, existieren. Es sollte beachtet werden, dass an beiden Standorten zwei Benutzer vorhanden sein müssen, d. h. der Benutzer SYSTEM und der Benutzer DBTOJMS. In Kapitel 6, „Sicherheit für alternative Clients”, auf Seite 25 finden Sie weitere Details zu diesen Benutzern.

In 10.7, „Konfigurieren der ausschließlichen Authentifizierung nach Identität”, auf Seite 38 finden Sie Details zur Konfiguration der ausschließlichen Authentifizierung nach Identität für einen Anwendungsserver.

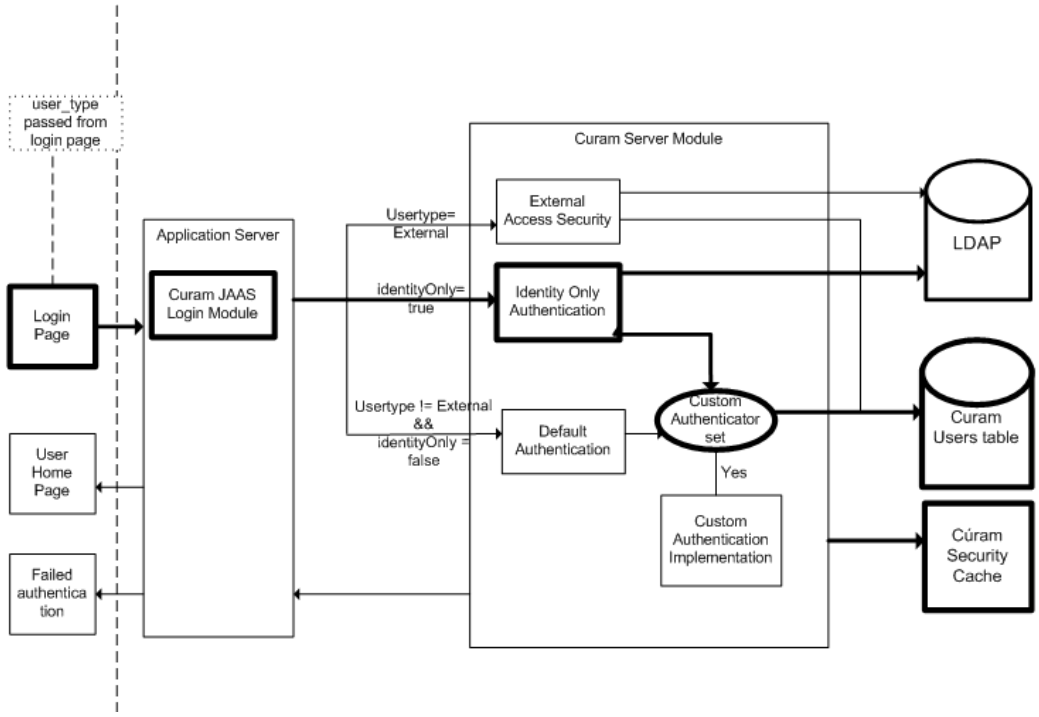

Abbildung 5. Ausschließliche Authentifizierung nach Identität

2.19 Anpassung der ausschließlichen Authentifizierung nach Identität

Die ausschließliche Implementierung nach Identität kann nicht angepasst werden; es ist jedoch möglich, weitere Überprüfungen durch Implementieren des angepassten Authentifikators hinzuzufügen. In 2.21, „Angepasste Überprüfungen" finden Sie weitere Details.

2.20 Externer Zugriff - Sicherheitsauthentifizierung

Die Architektur ermöglicht es einem Anwendungsentwickler, seine eigene angepasste Authentifizierungslösung für externe Benutzer zu implementieren; dabei wird für die vorhandene Authentifizierungs- und Berechtigungsinfrastruktur von SDEJ ein „Hook" bereitgestellt.

Um die angepasste Lösung in der Anwendung zu „verankern", muss die Klasse `curam.util.security.PublicAccessUser` erweitert werden. Dazu ist die Implementierung der Schnittstelle `curam.util.security.ExternalAccessSecurity` erforderlich. Diese Klasse wird während des Authentifizierungs- und Berechtigungsprozesses verwendet, um erforderliche Informationen zu dem externen Benutzer zu ermitteln. In Kapitel 13, „Anpassen von Anwendungen für externe Benutzer", auf Seite 51 finden Sie weitere Details.

2.21 Angepasste Überprüfungen

Es gibt Unterstützung für das Hinzufügen angepasster Überprüfungen zum Authentifizierungsprozess; so kann z. B. ein Benutzer dazu aufgefordert werden, eine Sicherheitsfrage zu beantworten, die anschließend überprüft werden muss. Der angepasste Code wird, sofern er implementiert ist, nach den erforderlichen IBM Cúram Social Program Management-Überprüfungen oder der Identitätszusicherung aufgerufen, und zwar erst, wenn diese erfolgreich waren.

Nach dem Aufrufen der angepassten Überprüfungen werden durch den Authentifizierungsprozess die relevanten Felder in der Benutzerdatenbanktabelle aktualisiert.

In 10.5, „Hinzufügen von angepassten Überprüfungen zum Authentifizierungsprozess", auf Seite 38 finden Sie weitere Details.

Kapitel 3. Berechtigung

3.1 Übersicht

In IBM Cúram Social Program Management wird der Prozess der Erteilung oder Verweigerung eines Benutzerzugriffs auf Funktionselemente einer Anwendung "Berechtigung" genannt. Ein Funktionselement kann jedes Element sein, an das eine eindeutige ID angehängt werden kann, z. B.:

- Serverprozessaufruf
- Element der Anwendung, für das eine Sicherheitsprüfung erforderlich ist, z. B. eine Reihe registrierter Fürsorgeprodukte.

Der Zugriff auf das Funktionselement wird durch eine Sicherheits-ID (SID) gesteuert, die einen Teil der IBM Cúram Social Program Management-Berechtigungsdaten bildet. Diese Daten werden mit einem Benutzer verknüpft und können mithilfe der Cúram-Administrationsanzeigen oder über den Data Manager konfiguriert werden. Weitere Details finden Sie im Handbuch *Cúram Server Developer*.

Die für die Berechtigung erstellten Sicherheitsdaten sind für die Verarbeitung wichtig, die bei jedem Client/Server-Aufruf durchgeführt wird; aus Leistungsgründen ist es außerdem wichtig, dass der Zugriff optimiert ist. Der Cúram-Sicherheitscache ist für das Caching von Berechtigungsdaten für einen Benutzer zuständig. In 5.2, „Cúram-Sicherheitscache”, auf Seite 23 finden Sie weitere Details.

In den folgenden Abschnitten wird die Beziehung dieser Berechtigungskonzepte und deren Funktion in IBM Cúram Social Program Management beschrieben.

3.2 Benutzer, Rollen und Gruppen

Die einer Anwendung zugeordneten Sicherheitsinformationen müssen zunächst in Sicherheitsprofilen zusammengefasst werden, bevor sie in einer Laufzeitumgebung verwendet werden können. Ein Sicherheitsprofil besteht aus einer Sicherheitsrolle, mindestens einer Sicherheitsgruppe sowie Zuordnungen zwischen Sicherheits-IDs (SIDs) und sicherbaren Elementen einer Anwendung.

Jedem berechtigten Benutzer wird während der Sicherheitskonfiguration eine Sicherheitsrolle zugewiesen; diese Rollen werden einer Reihe von Sicherheitsgruppen zugewiesen. Jeder Sicherheitsgruppe wird einer Reihe von Sicherheits-IDs zugeordnet. Die Sicherheits-ID stellt die sicherbaren Elemente von IBM Cúram Social Program Management dar, z. B. eine Methode oder ein Feld. Die Rollen-, Gruppen- und ID-Informationen werden in der Datenbank in einer Reihe von Tabellen gespeichert und über den Data Manager der Anwendung oder die Cúram-Administrationsanzeigen konfiguriert.

Durch diese Datenstruktur kann jeder Benutzer für ein beliebiges gesichertes Element einer Anwendung berechtigt werden. Dies ist eine leistungsfähige und flexible Möglichkeit für die Bereitstellung der Berechtigung für Cúram-Benutzer.

Es gibt eine Gruppe mit einer Mindestanzahl von SIDs, die erforderlich sind, damit ein Benutzer mit der Social Program Management Platform-Anwendung arbeiten kann. Diese SIDs werden der Standardgruppe BASESECURITYGROUP zugeordnet. Zur Angabe der Liste dieser SIDs können Sie die Datei `EJBServer/components/core/data/initial/handcraftedscripts/Supergroup.sql` zurate ziehen. Diese Datei ist für die Verknüpfung der SIDs mit der Standardgruppe BASESECURITYGROUP zuständig.

Eine einfache Methode, um sicherzustellen, dass alle Benutzer über Zugriffsrechte für diese Gruppe von SIDs verfügen, ist die Erstellung einer einzelnen Sicherheitsgruppe für diese Benutzer, die dann jeder einzelnen Sicherheitsrolle im System zugeordnet wird.

3.3 Sicherheits-IDs (SIDs)

Jedem geschützten Element in IBM Cúram Social Program Management wird eine SID zugewiesen, die in der gesamten Anwendung eindeutig ist.

Der Berechtigungsprozess ist in der Infrastruktur integriert; sobald die sicherbaren Elemente ermittelt worden sind, werden die verbliebenen Elemente von Codegeneratoren, Scripts und den Cúram-Administrationsanzeigen verarbeitet. Die Analyse der Elemente, die sicherbar sein müssen, ist ein manueller Prozess, der vom Anwendungsentwickler oder dem Sicherheitsadministrator durchgeführt werden muss. In diesem Abschnitt wird die für die Einrichtung der Berechtigung verfügbare Infrastruktur beschrieben.

Der erste zu berücksichtigende Berechtigungstyp ist die Prozessmethode (Fassade), die auch als *Sicherheit auf Funktionsebene* bezeichnet wird. Im Cúram-Modell kann ein Anwendungsentwickler wählen, ob die Sicherheit auf Prozessmethodenebene ein- oder ausgeschaltet wird. Die Option gilt nur für Geschäftsprozessobjekte, da sie die dem Client verfügbar gemachten Aufrufe einbinden. Entitätsobjektmethoden sind am Berechtigungsprozess nicht beteiligt.

Es gibt eine Reihe von SID-Typen; dies sind die folgenden:

- Funktions-IDs (FIDs)
- Sicherheits-IDs auf Feldebene
- Benutzerdefinierte SID-Typen

3.4 Funktions-IDs (FIDs)

Funktions-IDs (FIDs) sind ein spezieller Typ von Sicherheits-ID (SID), bei der der Typ auf FUNCTION gesetzt ist. Wenn auf eine Methode öffentlich zugegriffen werden kann (indem das Stereotyp in dem Modell auf "facade" gesetzt wird), wird eine FID für diese Methode generiert und die Sicherheit wird automatisch eingeschaltet.

Es ist möglich, die Sicherheit für eine Prozessmethode zur Entwicklungszeit auszuschalten. In 11.11, „Ausschalten der Sicherheit für eine Prozessmethode”, auf Seite 43 finden Sie weitere Details hierzu.

3.5 Sicherheits-IDs auf Feldebene

Mit der Sicherheits-ID auf Feldebene kann die Berechtigung auf bestimmte Felder einer öffentlich zugänglichen Methode angewendet werden. Wenn ein Benutzer zur Laufzeit keine Zugriffsrechte zum Anzeigen des anzuzeigenden Felds aufweist, wird der Inhalt des Felds als eine Reihe von Sternen (***) angezeigt. Weitere Informationen zu Sicherheits-IDs auf Feldebene finden Sie im Referenzhandbuch zu *Cúram Modeling*.

3.6 Benutzerdefinierte SIDs

In den vorherigen Abschnitten wurde Folgendes beschrieben:

Funktions-IDs (FIDs)
Automatisch generierte Sicherheits-ID (SID) des Typs "Funktion".

Sicherheits-IDs (SIDs) auf Feldebene
Auf bestimmte Felder einer Methode angewendete Sicherheit.

Darüber hinaus gibt es das Konzept der benutzerdefinierten SID. Der Berechtigungsprozess ist flexibel genug, um sämtliche zu sichernden Elemente einer IBM Cúram Social Program Management-Anwendung aufzunehmen. Der Anwendungsentwickler kann den Berechtigungsprozess wirksam anpassen, indem er neue *Typen* von SIDs definiert. Die neuen Typen stellen ein konzeptionelles Element mit Sicherheitsbedarf dar. Mit der folgenden Serverschnittstellenmethode kann die Berechtigung direkt für diese neuen benutzerdefinierten SID-Typen aufgerufen werden.

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

Standardmäßig sind die SIDs "LOCATION" und "PRODUCT" SIDs dieses Typs. Mit der oben stehenden Methode gibt es im Endeffekt keine Einschränkung für die SID-Typen, die definiert werden können. In 11.14, „Berechtigen neuer SID-Typen”, auf Seite 43 finden Sie weitere Details.

3.7 Laufzeitberechtigung

Über die IBM Cúram Social Program Management-Infrastruktur werden Berechtigungsprüfungen sowohl web-client-seitig als auch web-server-seitig durchgeführt.

3.8 Clientberechtigungsprüfungen

Bevor ein Benutzer auf eine Methode oder ein Feld zugreifen kann, führt der Web-Client vor dem ersten Laden der Seite Berechtigungsprüfungen durch. Wenn der Benutzer keine Zugriffsberechtigung hat, schlägt die Clientberechtigungsprüfung fehl und der Server wird nicht aufgerufen. Diese Prüfung kann in der Datei `curam-config.xml` konfiguriert werden, indem die Eigenschaft SECURITY_CHECK_ON_PAGE-_LOAD festgelegt wird. Weitere Details hierzu finden Sie in Abschnitt 3.12.13 zur allgemeinen Konfiguration im Referenzhandbuch zu *Cúram Web Client*.

Standardmäßig werden solche Berechtigungsfehler für den Web-Client nicht aufgezeichnet. Dieses Verhalten ist konfigurierbar. In 11.13, „Steuern der Protokollierung von Berechtigungsfehlern für den Client”, auf Seite 43 finden Sie weitere Details.

3.9 Serverberechtigungsprüfungen

Für die Bereitstellung von weiterem Zugriff auf IBM Cúram Social Program Management und die Position für die Inaktivierung der Berechtigungsprüfung über den Web-Client gibt es eine Berechtigungsprüfung auf der zweiten Ebene, die vom Server durchgeführt wird. Bei dieser serverseitigen Prüfung werden Berechtigungsfehler stets protokolliert; die Eigenschaft "client" wirkt sich nicht auf diese Protokollierung aus.

Das Protokoll sämtlicher Berechtigungsfehler wird in der Datenbank gespeichert, damit diese Fehler zu einem späteren Zeitpunkt geprüft werden können. In der Tabelle "AuthorisationLog" sind der Benutzername und die Sicherheits-ID für die fehlgeschlagene Berechtigung sowie eine Zeitmarke gespeichert, die angibt, wann der Fehler aufgetreten ist. In 11.15, „Analysieren der Datenbanktabelle "AuthorisationLog"”, auf Seite 44 finden Sie weitere Details zur Tabelle "AuthorisationLog".

Kapitel 4. Verschlüsselung in Cúram

4.1 Übersicht

Die in IBM Cúram Social Program Management verwendete Verschlüsselung bezieht sich im Allgemeinen auf zwei Arten von Funktionen für die Sicherheit Ihrer Cúram-Systeme:

1. Chiffrierung - Zwei-Weg-Verschlüsselung von Kennwörtern, die zu unterschiedlichen Verabeitungszeitpunkten verwendet wird.
2. Digest-Verschlüsselung - Ein-Weg-Hashing (oder Digesting) von Kennwörtern, z. B. bei der Anmeldung.

Die Werte zum Konfigurieren des kryptografischen Verhaltens können vom Benutzer über die Eigenschaftendatei (`CryptoConfig.properties`) ausgewählt werden, damit Ihre Cúram-Installation die höchstmögliche Steuerung und Sicherheit erhält. Durch diese Flexibilität ist es möglich, das System an sich ändernde Sicherheitsstandards anzupassen. Informationen, wie eine Verschlüsselung konfiguriert und angepasst werden kann, finden Sie in Kapitel 12, „Anpassen der Verschlüsselung", auf Seite 45.

Für bestehende Benutzer von IBM Cúram Social Program Management wird empfohlen, dass die Systemkennwörter (neue Chiffrierung) und Benutzerkennwörter (neuer Digest) aus den vorhandenen OOTB-Standardwerten aktualisiert werden, um die Sicherheit zu erhöhen. Da das Aktualisieren von Benutzerkennwörtern eine erhebliche Änderung darstellt, können Sie diese beiden Änderungen unabhängig voneinander durchführen (siehe Beschreibung in den zugehörigen Themen). Wenn Sie eine geringere Sicherheitsstufe akzeptieren, können Sie auf eigenes Risiko das vorhandene System und die Benutzerkennwörter unverändert lassen. Dies wird jedoch nicht empfohlen.

Folgende Verschlüsselungskonfigurationen werden unterstützt:

1. AES: 128, 192, 256 (konform mit FIPS 140-2 und SP800-131a)
2. Two-Key-Triple-DES - DESede: 112 (konform mit FIPS 140-2)
3. Three-Key-Triple-DES - DESede: 168 (konform mit FIPS 140-2 und SP800-131a)
4. Keine Verschlüsselungskonfiguration, die durch Entfernen der Datei `CryptoConfig.properties` festgelegt wird. In diesem Fall wird Cúram auf die vorherigen OOTB-Verschlüsselungseinstellungen zurückgesetzt.

In der Umgebung, in der Cúram ausgeführt wird, haben der Anwendungsserver, die Datenbank und weitere Software (z. B. Web-Server, LDAP usw.) ihre eigene kryptografische Unterstützung. Informationen dazu finden Sie in den entsprechenden Anbieterdokumentationen.

4.2 Chiffrierung

Eine Chiffrierung bezieht sich auf das Verschlüsseln von Kennwörtern, d. h., es handelt sich um einen Zwei-Wege-Prozess, bei dem entschlüsselbare Werte dargestellt werden. Es gibt einige Dutzend dieser verschlüsselten Kennwörter in verschiedenen Eigenschaftendateien in Cúram und das Verschlüsseln dient dazu, sie zu schützen. Zu einem bestimmten Nutzungszeitpunkt werden sie entschlüsselt, z. B. bei der Verbindung zu Ihrem Datenbanksystem.

4.3 Digest-Verschlüsselung

Eine Digest-Verschlüsselung bezieht sich auf die Behandlung von Kennwörtern, die nicht entschlüsselt werden müssen, sondern auf das Speichern von Kennwörtern für einen späteren Vergleich, z. B. Cúram-Benutzeranmeldungen. Das heißt, es handelt sich um einen Ein-Wege-Prozess, bei dem nicht entschlüsselbare Werte dargestellt werden.

4.4 Verschlüsselungseigenschaften

Die Cúram-Datei `CryptoConfig.properties` enthält Einstellungen für die Chiffrierung und Digest-Verschlüsselung. Desahlb sollten diese Datei sowie alle Dateien, auf die sich die Datei bezieht (z. B. Keystore und Salt), als kritische Elemente für Ihre Systemsicherheit betrachtet und mit entsprechenden Zugriffskontrollen (z. B. Dateiberechtigungen) versehen sowie für die Nutzung für Produktionssysteme speziell modifiziert und isoliert werden. Das heißt, wenn die Details der Datei allgemein bekannt werden, obwohl dies möglicherweise kein Sicherheitsrisiko selbst darstellt, ist eine Sicherheitsstufe weniger vorhanden und kann eine Verschlüsselungsänderung notwendig machen, die die Verarbeitung unterbricht (siehe 12.2, „Chiffrierungsanpassung", auf Seite 45 und 12.5, „Digest-Anpassung", auf Seite 47).

Zugehörige Themen:

- 4.5, „Cúram-Chiffrierungseigenschaften"
- 4.6, „Cúram-Digest-Eigenschaften", auf Seite 21

4.5 Cúram-Chiffrierungseigenschaften

In den Cúram-Eigenschaftendateien und -konfigurationen werden verschiedene Kennwörter in einem verschlüsselten OOTB-Format gespeichert.

Die Cúram-Verschlüsselungskonfiguration ist sofort einsatzfähig (OOTB, out-of-the-box). Es wird jedoch empfohlen, diese Einstellungen in Hinblick auf Ihre lokalen Sicherheitsanforderungen zu modifizieren. So sind die OOTB-Einstellungen in Entwicklungsumgebungen möglicherweise gut geeignet, es wird jedoch dringend empfohlen, in Produktionsumgebungen diese Einstellungen zu modifizieren (z. B. den geheimen Chiffrierungsschlüssel zu ändern).

Die Chiffrierungseinstellungen sind in der Datei `CryptoConfig.properties` gespeichert. Die Eigenschaften und ihre Werte lauten wie folgt:

- `curam.security.crypto.cipher.algorithm`
  - **Gültige Werte:** in der JCE-Dokumentation beispielsweise: http://docs.oracle.com/javase/6/docs/technotes/guides/security/StandardNames.html#Cipher. Die unterstützte Chiffrierung sind AES und verschiedene Formen von Triple-DES.
  - **Standard:** AES (konform mit FIPS 140-2 und SP800-131a)
- `curam.security.crypto.superseded.cipher.algorithm`
  - **Gültige Werte:** siehe `curam.security.crypto.cipher.algorithm`
  - **Standard:** keiner
  - **Zweck:** für höhere Flexibilität zur Unterstützung der Aktualisierungs-/Migrationsphase mit angepasstem Code (z. B. ein Stapelverarbeitungsprogramm) über die API `curam.util.security.EncryptionUtil.decryptSupersededPassword()`.
- `curam.security.crypto.cipher.keystore.location`
  - **Gültige Werte:** Pfad zur Keystore-Datei, die den geheimen Schlüssel enthält. Dies kann eine Angabe zu einem absoluten Pfad sein oder ein Pfad relativ zum Klassenpfad (z. B. `CuramSample.keystore`).
  - **Standard:** keiner
- `curam.security.crypto.cipher.keystore.storepass`
  - **Gültige Werte:** gemäß JDK-Befehl **`keytool`**.
  - **Standard:** Kennwort
  - **Zweck:** Angabe des Kennworts für den Zugriff auf den Keystore.
- `curam.security.crypto.cipher.provider.class`
  - **Gültige Werte:** vollständig qualifizierter Name der JCE-Verschlüsselungsproviderklasse.
  - **Standard:** leer

– **Zweck:** optionale Möglichkeit, die Verwendung eines alternativen, auf Standards basierenden Provider zu aktivieren.

Diese Chiffrierungsfunktion gilt für die Eigenschaften, die in 4.7, „Per Chiffrierung verschlüsselte Kennwörter”, auf Seite 22 beschrieben werden.

Diese Cúram-Verschlüsselungseinstellungen werden standardmäßig als "sofort einsatzfähig" (OOTB) aktiviert und stellen Änderungen dar, die Cúram-Installationen berücksichtigen müssen (siehe Handbuch *Cúram Upgrade*).

4.6 Cúram-Digest-Eigenschaften

Interne und externe Cúram-Benutzer, die über eine ausschließliche Authentifizierung nach Identität nicht aufgerufen werden, werden anhand einer formularbasierten Anmeldung authentifiziert. Das in das Formular eingegebene Kennwort wird mittels Digest verschlüsselt und mit dem Digest-Wert verglichen, der in der Datenbank für den Benutzer gespeichert wurde.

Anmerkung: Diese Verarbeitung wird nicht auf Benutzer angewendet, die in Drittanbietersystemen wie LDAP authentifiziert werden.

Die Cúram-Verschlüsselungskonfiguration ist sofort einsatzfähig (OOTB, out-of-the-box). Es wird jedoch empfohlen, diese Einstellungen in Hinblick auf Ihre lokalen Sicherheitsanforderungen zu modifizieren. So sind die OOTB-Einstellungen in Entwicklungsumgebungen möglicherweise gut geeignet, es wird jedoch dringend empfohlen, in Produktionsumgebungen diese Einstellungen zu modifizieren (z. B. verschlüsselter Digest-Salt-Wert).

Die Digest-Einstellungen sind in der Datei `CryptoConfig.properties` gespeichert. Die Eigenschaften und ihre Werte lauten wie folgt:

- `curam.security.crypto.digest.algorithm`
  - **Gültige Werte:** in der JCE-Dokumentation beispielsweise http://docs.oracle.com/javase/6/docs/technotes/guides/security/StandardNames.html#MessageDigest. Die unterstützten Digests sind SHA-Varianten (1, 256 usw.) und MD5.
  - **Standard:** SHA-256 (konform mit FIPS 140-2 und SP800-131a)
  - **Zweck:** Spezifikation des Digest-Algorithmus.
- `curam.security.crypto.digest.salt.location`
  - **Gültige Werte:** ein Pfad, der die Datei identifiziert, die den verschlüsselten, geheimen Digest-Salt enthält.
  - **Standard:** keiner
  - **Zweck:** eine optionale Datei zur Angabe des Salt (verschlüsselt) für die Digest-Verschlüsselung.
- `curam.security.crypto.digest.iterations`
  - **Gültige Werte:** 0 eine positive Ganzzahl.
  - **Standard:** 0
  - **Zweck:** Höhere Werte bedeuten in der Regel höhere Sicherheit, aber zu Lasten der Verarbeitung (z. B. bei der Anmeldezeit).

Es gibt eine Reihe von "ersetzten" Eigenschaften, die beim Migrieren von einem Satz von Digest-Einstellungen oder Standards auf den anderen Satz eine höhere Flexibilität erlauben. Folgendes hat eine ähnliche Funktion wie ihre Entsprechung oben, aber die werden von der Cúram-Verschlüsselungsfunktion zur Unterstützung alter und neuer Einstellung für die Migrationsphase unterstützt:

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

Die Verwendung und das Verhalten der ersetzten Eigenschaften wird von der Eigenschaft curam.security.convertsupersededpassworddigests.enabled gesteuert, die von der Benutzerschnittstelle der Eigenschaftenverwaltung verwaltet werden. Weitere Informationen zur Verwendung ersetzter Eigenschaften finden Sie in 12.7, „Vorgehensweise beim Verwenden der ersetzten Digest-Einstellungen für einen Migrationszeitraum”, auf Seite 48.

4.7 Per Chiffrierung verschlüsselte Kennwörter

Die Kennwörter werden in Cúram per Chiffrierung verschlüsselt:

- Bootstrap.properties:
  - curam.db.password - Datenbankkennwort
  - curam.searchserver.sync.password - weitere Informationen siehe *Cúram-Server für generische Suche*
- AppServer.properties: (In der Regel wird die Eigenschaftendatei zum Konfigurieren von Testservern verwendet und eignet sich nicht für Produktionssysteme.)
  - security.password - Kennwort für die Adminstrationskonsole des Anwendungsservers
  - curam.security.credentials.async.password - ersetzt runas.password property
- Application.prx - einzelne Eigenschaftsbeschreibungen werden mit den Eigenschaften in der Benutzerschnittstelle der Cúram-Eigenschaftsverwaltung dokumentiert:
  - curam.security.credentials.dbtojms.password - (in Verbindung mit "curam.security.credentials.dbtojms.username"), ersetzt die Schnittstellen-APIs "curam.omega3.DBtoJMSCredentialsIntf", die zuvor zum Bereitstellen der angepassten Berechtigungsnachweise für DB-TO-JMS verwendet wurden
  - curam.security.credentials.ws.password (in Verbindung mit curam.security.credentials.ws.username), ersetzt die Standardeinstellungen der Berechtigungsnachweise für die Buildzeit-Standard-Web-Services
  - curam.meeting.request.reply.password - (ein SMTP-Kennwort)
  - curam.ldap.password
  - curam.citizenworkspace.password.protection.key
- BIBootstrap.properties - nur BIRT-Benutzer, siehe Handbuch *Cúram Business Intelligence BIRT Developer*:
  - curamsource.db.password
  - central.db.password
  - centraldm.db.password
- Web-Services - siehe :
  - ws_inbound.xml - <ws_service_password>
  - services.xml - <parameter name="jndiPassword">
- CTM - Cúram Transport Manager:
- Die Spalte "Password" der Tabelle "TargetSystemService" enthält ein verschlüsseltes Kennwort.

Kapitel 5. Caching von Sicherheitsdaten

5.1 Übersicht

In diesem Kapitel wird der Cúram-Sicherheitscache beschrieben, in dem sämtliche Berechtigungsdaten für einen Benutzer gespeichert werden. Details zum WebSphere-Cache und zu dessen Bedeutung für die Authentifizierung eines Benutzers bei der Anmeldung werden ebenfalls in diesem Kapitel beschrieben.

5.2 Cúram-Sicherheitscache

Die Sicherheitsinformationen aus den Datenbanktabellen, die die in 3.2, „Benutzer, Rollen und Gruppen", auf Seite 15 beschriebenen Profile unterstützen, werden von der Infrastruktur in den Cache gestellt. Dieser Vorgang dient der Optimierung der Suche nach und dem Abrufen von Daten während des Berechtigungsprozesses.

Zur Optimierung der Leistung wird der Cache bei Bedarf geladen, wenn es zu Sicherheitsberechtigungsanforderungen für die Anwendung kommt; der Cache ist eine gemeinsam genutzte Ressource. Im Hinblick auf Anwendungscode ist der Cache eine geschützte Ressource, auf die nicht direkt zugegriffen werden kann. Der Zugriff ist (ausschließlich für Abfragen) über die Berechtigungsschnittstelle `curam.util.security.Authorisation` möglich, die einem Entwickler die Implementierung einer angepassten Berechtigungsprozedur ermöglicht. In 11.14, „Berechtigen neuer SID-Typen", auf Seite 43 finden Sie weitere Details hierzu.

Wenn die Eigenschaft "curam.security.casesensitive" auf "false" gesetzt ist, werden im Sicherheitscache alle Benutzernamen in Großbuchstaben gespeichert und bei allen Cacheabfragen werden die Buchstaben des angegebenen Benutzernamens automatisch in Großbuchstaben geändert. Es sollte auch beachtet werden, dass falls doppelte Benutzernamen ohne Beachtung der Groß-/Kleinschreibung vorhanden sind, dies bei der Initialisierung des Sicherheitscache zu einem schwerwiegender Fehler führt. In 10.4, „Ändern der Groß-/Kleinschreibung des Benutzernamens", auf Seite 37 finden Sie weitere Details hierzu.

5.3 Cacheaktualisierung

Da Sicherheitsdaten für den Betrieb von IBM Cúram Social Program Management äußerst wichtig sind, muss der Cache immer dann aktualisiert werden, wenn Änderungen an den sicherheitsrelevanten Datenbanktabellen vorgenommen wurden. Die Aktualisierung des Cúram-Sicherheitscache ist ein asynchroner Prozess.

5.4 Cacheaktualisierungsfehler

Die Aktualisierung des Cúram-Sicherheitscache wird entweder durch einen Anwendungswarmstart oder über die Cúram-Administrationsanzeigen durch den Systemadministrator (sysadmin) ausgelöst; aus diesem Grund erhält der Administrator kein Feedback, wenn das erneute Laden des Cache fehlschlägt. Wenn nach einer Aktualisierung deren Erfolg durch Überprüfen der Systemprotokolle oder manuelles Prüfen der Anwendung sichergestellt werden soll, kann dies eine lästige Aufgabe sein. Es ist daher empfehlenswert, die optionale Callback-Schnittstelle für ein Feedback im Fall eines Fehlers beim erneuten Laden des Cache zu implementieren. In 10.8, „Hinzufügen der Callback-Schnittstelle für fehlerhafte Cacheaktualisierung", auf Seite 38 finden Sie weitere Details.

5.5 Caching-Verhalten von WebSphere

Bei WebSphere werden Benutzerinformationen und Berechtigungsnachweise in den eigenen Sicherheitscache gestellt. Das Cúram-Anmeldemodul wird nicht aufgerufen, solange ein Benutzereintrag in diesem Cache gültig ist. Die Standardzeit für die Invalidierung dieses Sicherheitscache beträgt zehn Minuten; dies bedeutet, dass der Benutzer zehn Minuten lang inaktiv war.

Wenn sich beispielsweise ein Benutzer über den Web-Client zum ersten Mal bei der Anwendung anmeldet, wird er zur Eingabe seines Benutzernamens und seines Kennworts aufgefordert. Daraufhin wird das Cúram-Anmeldemodul aufgerufen und authentifiziert die angegebenen Informationen. Wenn dieser Benutzer einen zweiten neuen Web-Browser öffnet und versucht, auf die Anwendung zuzugreifen, wird er erneut dazu aufgefordert, seinen Benutzernamen und sein Kennwort einzugeben. Wenn WebSphere diese Informationen empfängt, wird der Sicherheitscache abgefragt, um festzustellen, ob sich der Benutzername und das Kennwort bereits im Cache befinden. Falls dies der Fall ist und das Kennwort übereinstimmt, fragt WebSphere die Anmeldemodule nicht ab.

Bei diesem Verhalten wirken sich sämtliche Änderungen an den Kontoeinschränkungen eines Benutzers oder am Kennwort erst aus, wenn der Benutzer im Sicherheitscache von WebSphere ungültig gemacht wurde.

Weitere Informationen finden Sie im entsprechenden *WebSphere Application Server Information Center*.

Kapitel 6. Sicherheit für alternative Clients

6.1 Übersicht

Es gibt Prozesse, die möglicherweise keinem bestimmten angemeldeten Benutzer zugeordnet werden können. Dazu gehören alternative Clients, z. B. Nicht-Webprozesse wie Batchverarbeitung, Web-Services und verzögerte Verarbeitung. Da jeder Prozess, der mit einer IBM Cúram Social Program Management-Anwendung interagiert, authentifiziert werden muss, muss für jeden dieser Prozesse ein gültiger Benutzer vorhanden sein. In den folgenden Abschnitten sind Details zu den Benutzern aufgeführt, die in der Cúram-Benutzertabelle vorhanden sein müssen, sowie zu den Prozessen, die von diesen Benutzern abhängig sind.

6.2 Obligatorische Cúram-Benutzer

In der Cúram-Benutzerdatenbanktabelle muss immer eine Reihe von Benutzern vorhanden sein. Diese Benutzer sind für Anwendungsprozesse wie verzögerte Verarbeitung und Workflow erforderlich. Sind diese Benutzer nicht vorhanden, schlägt die Authentifizierung fehl, woraufhin auch diese Prozess fehlschlagen.

Die unten stehenden Benutzernamen und Kennwörter für die Prozesse sind Standardberechtigungsnachweise; es ist empfehlenswert, diese Berechtigungsnachweise aus Sicherheitsgründen zu ändern.

Dazu zählen die folgenden Benutzer:

- SYSTEM

  Bei dem Benutzer SYSTEM handelt es sich um den Benutzer, der für die Ausführung von JMS-Nachrichten zuständig ist. Dieser Benutzer muss existieren; für den Benutzernamen muss die Groß-/Kleinschreibung beachtet werden. In 6.5, „JMS-Messaging", auf Seite 26 finden Sie weitere Details.
- DBTOJMS

  Bei dem Benutzer DBTOJMS handelt es sich um den Standardbenutzer, der für die Ausführung des DBToJMS-Triggers für die Batchverarbeitung zuständig ist. Dieser Benutzer muss existieren; für den Benutzernamen muss die Groß-/Kleinschreibung beachtet werden. In 6.4, „Batchverarbeitung", auf Seite 26 finden Sie weitere Details.
- WEBSVCS

  Bei dem Benutzer WEBSVCS handelt es sich um den Standardbenutzer, der für die Ausführung von Web-Services zuständig ist. Dieser Benutzer muss existieren; für den Benutzernamen muss die Groß-/Kleinschreibung beachtet werden. In 6.3, „Web-Services" finden Sie weitere Details.

6.3 Web-Services

Für Apache Axis2 (empfohlene Implementierung für Web-Services) gibt es für die Authentifizierung Standardberechtigungsnachweise. Ein Benutzer hat die Möglichkeit, diese Berechtigungsnachweise bei Bedarf global zu ändern oder für jeden Service separat. Um sicherzustellen, dass Web-Services keinen Sicherheitsverstößen ausgesetzt sind, ist dieser Standardbenutzer standardmäßig nicht dazu berechtigt, auf Web-Services zuzugreifen. Für die Berechtigung muss ein Web-Service einer Sicherheitsgruppe und wiederum einer Sicherheitsrolle zugeordnet werden, die mit dem Benutzer (z. B. WEBSVCS) für den Zugriff verknüpft wird. Um sicherzustellen, ob der Benutzer berechtigt ist, ist ein manueller Prozess erforderlich. Im Abschnitt zur Anpassung der Laufzeitfunktion für Empfänger im Handbuch *Cúram Web Services* finden Sie weitere Details zu Web-Services; lesen Sie auch das Kapitel zur Berechtigung im vorliegenden Handbuch.

Bei Apache Axis 1.4 (traditionelle Web-Services) wird dieser Web-Service unter Verwendung der Standardberechtigungsnachweise automatisch bei der Anwendung angemeldet, sobald ein Prozess als Web-

Service modelliert wird. Dieser Standardbenutzer wird für die Berechtigung automatisch eingerichtet, d. h. der Benutzer kann auf den erstellten Web-Service zugreifen. Aus diesem Grund ist Vorsicht geboten, wenn eine Klasse als Web-Service sichtbar gemacht wird. Weitere Informationen finden Sie im Abschnitt zu den traditionellen eingehenden Web-Services im Handbuch *Cúram Web Services*.

Es gibt eine Reihe weiterer Themen, die sich auf die Sicherheit von Web-Services beziehen, die Rampart verwenden, z. B. das Thema Datenverschlüsselung. Weitere Details hierzu finden Sie im Handbuch *Cúram Web Services*.

6.4 Batchverarbeitung

Da der Anwendungsserver für das Batchstartprogramm nicht aktiv sein muss, führt dieses auch keine Authentifizierung oder Berechtigung auf Anwendungsebene durch. Es ist lediglich eine Authentifizierung für die Datenbank erforderlich. Das Batchstartprogramm verwendet für die Herstellung einer Verbindung zu der Datenbank und die Ausführung von Batchprogrammen dieselben Berechtigungsnachweise wie der Anwendungsserver (im Verzeichnis `%SERVER_DIR%/project/properties/Bootstrap.properties`).

Das Batchstartprogramm bzw. die Batchprogramme können den Anwendungsserver optional auslösen, um einen DB-zu-JMS-Transfer zu starten. Dies beinhaltet die Anmeldung und den Aufruf einer Methode auf dem Server, für den wiederum ein gültiger Benutzername und ein gültiges Kennwort erforderlich sind. Standardmäßig werden für die DB-zu-JMS-Transferoperation Standardberechtigungsnachweise verwendet; aus diesem Grund muss das Konto DBTOJMS in der Cúram-Benutzertabelle vorhanden und aktiviert sein und ihm muss die Rolle SYSTEMROLE zugewiesen sein, damit die Berechtigung möglich ist. Die länderspezifischen Angaben für den DB-zu-JMS-Transfer sind die standardmäßigen länderspezifischen Angaben für diesen Benutzer; diese sind im Feld "defaultLocale" der Benutzertabelle angegeben.

Weitere Details zum Ändern des Benutzers für den DB-zu-JMS-Transfer finden Sie im Abschnitt zu den Sicherheitsaspekten im Handbuch *Cúram Batch Processing*.

Die Eigenschaft "batch.username" kann für die Angabe des Benutzernamens für die Operationen verwendet werden, die vom Batchstartprogramm ausgeführt werden. Für diese Eigenschaft wird der Parameter -D gesetzt. Beispiel: build runbatch -Dbatch.username=admin

6.5 JMS-Messaging

JMS-Nachrichten werden zu Kommunikationszwecken bei verzögerten Prozessen und Workflow verwendet. Da JMS-Nachrichten vom Anwendungsserver ausgelöst werden und mit der IBM Cúram Social Program Management-Anwendung interagieren müssen, müssen gültige Cúram-Berechtigungsnachweise existieren. Es muss das Benutzerkonto SYSTEM in der Cúram-Benutzertabelle vorhanden und aktiviert sein und ihm muss die Rolle SYSTEMROLE zugewiesen sein, damit die Berechtigung gewährleistet werden kann. Die Ländereinstellung für JMS-Nachrichten ist die Standardländereinstellung für diesen Benutzer; diese ist im Feld "defaultLocale" der Benutzertabelle angegeben.

Der Benutzername SYSTEM kann während oder nach der Implementierung der Anwendung geändert werden. Weitere Informationen zum jeweiligen Anwendungsserver finden Sie im Handbuch *Cúram Server Deployment*.

6.6 Verzögerte Verarbeitung

Bei einem verzögerten Prozess in IBM Cúram Social Program Management handelt es sich um eine Geschäftsmethode, die asynchron aufgerufen wird. Da verzögerte Prozesse mit der Anwendung interagieren, müssen gültige Cúram-Berechtigungsnachweise existieren. Es muss das Benutzerkonto SYSTEM in der Cúram-Benutzertabelle vorhanden und aktiviert sein und ihm muss die Rolle SYSTEMROLE zugewiesen sein, damit die Berechtigung gewährleistet werden kann. Die Ländereinstellung für verzögerte Prozesse ist die Standardländereinstellung für diesen Benutzer; diese ist im Feld "defaultLocale" der Be-

nutzertabelle angegeben. Im Fall des Offline-Komponententests für verzögerte Prozesse ist der Benutzername leer und die tatsächliche Ländereinstellung ist die Standardländereinstellung für den IBM Cúram Social Program Management-Server.

Kapitel 7. Anwendungen für externe Benutzer

7.1 Übersicht

Die sofort einsatzfähige IBM Cúram Social Program Management-Standardanwendung ist für interne Benutzer aktiviert. Bei internen Benutzern handelt es sich um Benutzer, die in der Cúram-Benutzerdatenbanktabelle existieren. Ein typischer interner Benutzer wäre ein Fallbearbeiter, der Forderungen für Beteiligte erstellt und verwaltet und über vollständigen Zugriff auf die Anwendung verfügt. Die Infrastruktur stellt eine Funktionalität für die Authentifizierung und Berechtigung dieser internen Benutzer bereit.

Darüber hinaus muss ein Benutzer gesicherten Zugriff auf Teile der IBM Cúram Social Program Management-Anwendung haben können. Solche Benutzer sind keine Angehörigen der Organisation. Ihr Zugang ist eingeschränkt. Diese Benutzer werden als externe Benutzer betrachtet; die Authentifizierung dieser Benutzer kann durch die Verwendung des bereitgestellten Hookpunkts für die Sicherheit bei externem Zugriff vollständig angepasst werden. Da externe Benutzer anders als interne Benutzer verarbeitet werden, ist für externe Benutzer eine spezielle Webanwendung erforderlich.

7.2 Anwendungen für externe Benutzer

Bei der Entwicklung einer Anwendung für einen externen Benutzer müssen folgende Komponenten implementiert werden:

- Eine Clientanwendung für externe Benutzer, d. h. eine separate EAR-Datei mit der Web-Client-Anwendung.
- Eine angepasste Datei vom Typ `logon.jsp`, bei der die externe Anwendung einen Parameter vom Typ "user_type" übergeben muss, der angibt, dass sich ein externer Benutzer anmeldet.
- Eine angepasste Klasse, die `curam.util.security.PublicAccessUser` erweitert, muss bereitgestellt werden. Dazu ist die Implementierung der Schnittstelle `curam.util.security.ExternalAccessSecurity` erforderlich. Diese abstrakte Klasse enthält Methoden, die für die Authentifizierung und Berechtigung eines externen Benutzers zuständig sind.

Es gibt sowohl interne als auch externe Benutzertypen. Es kann auch unterschiedliche Typen von externen Benutzern geben. Es gibt beispielsweise einen externen Benutzer des Typs PUBLIC, der einschränkten Zugriff auf eine externe Anwendung hätte. Es könnte aber auch einen weiteren externen Benutzer vom Typ PROVIDER geben, bei dem es sich um einen registrierten externen Benutzer handelt. Die Möglichkeit unterschiedlicher Typen von externen Benutzern bietet mehr Flexibilität innerhalb einer externen Anwendung, was wiederum eine differenziertere Regulierung der Authentifizierung des externen Benutzers auf der Grundlage des Typs von externem Benutzer ermöglicht.

7.3 Benutzerbereich

Es gibt zwei unterschiedliche Typen bzw. Bereiche von Benutzern in der IBM Cúram Social Program Management-Anwendung: interne und externe Benutzer. Für die Bestimmung eines Typs von Benutzer wird eine der folgenden Methoden angewendet:

- Durch den Cúram-Sicherheitscache

  Wenn der Benutzer im Cúram-Sicherheitscache existiert, wird angenommen, dass es sich um einen internen Benutzer handelt. Wenn der Benutzer nicht im Sicherheitscache existiert, wird angenommen, dass es sich um einen externen Benutzer handelt. In diesem Fall (Standardverhalten) müssen sowohl die internen als auch die externen Benutzernamen eindeutig sein.
- Durch die angepasste Schnittstelle `UserScope`

  Diese Möglichkeit besteht, wenn die angepasste Schnittstelle `UserScope` implementiert ist. Diese angepasste Schnittstelle hat Vorrang vor der Ermittlung des Benutzertyps mittels Überprüfung des Benut-

zers im Cúram-Sicherheitscache. Weitere Details finden Sie in 13.15, „Mithilfe der Schnittstelle "UserScope" feststellen, ob ein Benutzer ein interner oder externer Benutzer ist", auf Seite 59.

Wenn es sich um einen externen Benutzer handelt, wird zur Bestimmung der Benutzerrolle die Implementierung der Methode `curam.util.security.ExternalAccessSecurity.getSecurityRole()` anstelle der internen Sicherheitsrollen verwendet. In 13.8, „Berechtigen eines externen Benutzers", auf Seite 55 finden Sie weitere Details zu dieser Methode.

Zur Unterstützung alternativer Methoden, um festzustellen, ob es sich bei dem Benutzer um einen internen oder externen Benutzer handelt, kann die angepasste Schnittstelle `UserScope` verwendet werden. Weitere Details finden Sie in 13.15, „Mithilfe der Schnittstelle "UserScope" feststellen, ob ein Benutzer ein interner oder externer Benutzer ist", auf Seite 59.

7.4 Bereitstellung einer externen Anwendung

Bei der Bereitstellung einer Anwendung auf einem Anwendungsserver ist die Sicherheitskonfiguration für den Anwendungsserver für alle IBM Cúram Social Program Management-Anwendungen gültig, die für diese Instanz des Anwendungsservers bereitgestellt wurden. Deshalb ist Vorsicht geboten, wenn Sie die Bereitstellungsarchitektur für mehr als eine Anwendung in Betracht ziehen. Dies ist ein wichtiger Aspekt bei der Entscheidung darüber, ob sowohl eine interne als auch eine externe Anwendung für ein und dieselbe Instanz des Anwendungsservers bereitgestellt werden soll.

Die folgenden Aspekte sollten berücksichtigt werden:

- Wird die Identität nur für interne Benutzer verwendet?
- Wird ein alternativer Authentifizierungsmechanismus verwendet, z. B. LDAP?
- Werden sowohl interne als auch externe Benutzer durch LDAP authentifiziert?

Die Antworten auf diese Fragen beeinflussen die Einstellung der Eigenschaften des Anwendungsservers (d. h. Eigenschaften, die in der Datei `AppServer.properties` angegeben sind), die sich wiederum auf das Verhalten des JAAS-Anmeldemoduls von Cúram auswirken. Diese Betrachtungen beeinflussen auch die Implementierung der Klasse `curam.util.security.PublicAccessUser` und der Schnittstelle `curam.util.security.ExternalAccessSecurity` für externe Benutzer.

Durch die Eigenschaften des Anwendungsservers im JAAS-Anmeldemodul von Cúram ist eine differenziertere Steuerung der Authentifizierung der Benutzertypen möglich. Externe und interne Benutzer können in einer Situation, in der interne und externe Anwendungen auf ein und demselben Anwendungsserver bereitgestellt werden, unterschiedlich authentifiziert werden, ebenso wie unterschiedliche Typen von externen Benutzern. Dazu gehören unter anderem die folgenden Eigenschaften:

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

  Setzen Sie diese Eigenschaft auf eine durch Kommas getrennte Liste mit Benutzertypen, für die die Benutzerregistry des Anwendungsservers *nicht* abgefragt wird, d. h., die Implementierung innerhalb der Methode `PublicAccessUser.authenticate()` ist für die Authentifizierung des externen Benutzers dieses Typs verantwortlich. So kann beispielsweise LDAP als Benutzerregistry konfiguriert werden.
- curam.security.user.registry.enabled.types

  Setzen Sie diese Eigenschaft auf eine durch Kommas getrennte Liste mit Benutzertypen, für die die Benutzerregistry abgefragt wird, d. h., bei der Implementierung innerhalb der Methode `PublicAccessUser.authenticate()` muss der Benutzer nicht vollständig authentifiziert werden. Die Benutzerregistry ist für die Authentifizierung dieses Typs von externem Benutzer zuständig. So kann beispielsweise LDAP als Benutzerregistry konfiguriert werden; in diesem Fall kann LDAP für die Authentifizierung dieser Typen von externem Benutzer verantwortlich sein.

Diese Eigenschaften sind von der Implementierung der Klasse `curam.util.security.PublicAccessUser` und der Schnittstelle `ExternalAccessSecurity` abhängig.

Beachten Sie die folgenden beispielhaften Projektanforderungen:

- Ein interner Benutzer muss sich mit LDAP authentifizieren.
- Ein externer Benutzer des Typs "EXT_PUBLIC" muss sich mit IBM Cúram Social Program Management und nicht mit LDAP authentifizieren.
- Ein externer Benutzer des Typs "EXTERNAL" darf sich nur mit LDAP und nicht mit IBM Cúram Social Program Management authentifizieren.
- Sowohl die internen als auch die externen Anwendungen werden auf ein und derselben Anwendungsserverinstanz bereitgestellt.

Die folgenden Einstellungen können für das obige Beispiel verwendet werden:

- curam.security.check.identity.only auf `true` gesetzt
- curam.security.user.registry.disabled.types=EXT_PUBLIC

Es müssen nicht nur die Eigenschaften festgelegt werden, sondern die Erweiterung `PublicAccessUser` (und die Implementierung der Schnittstelle `curam.util.security.ExternalAccessSecurity`) muss die Logik zur Bereitstellung unterschiedlicher Typen externer Benutzer und zur Vorgehensweise bei deren Authentifizierung aufweisen.

Kapitel 8. Verwenden von Single Sign-on

8.1 Übersicht

Die Zahl der Anwendungen in einem Unternehmen führt häufig zu einer Zunahme an belegten Benutzernamen und Kennwörtern, was wiederum zu einer eingeschränkten Funktionalität für den Benutzer und zu zusätzlichem Wartungsaufwand führt. Eine Vielzahl von Benutzernamen und Kennwörtern beeinträchtigt auch die Sicherheit, da Benutzer entweder einfache Kennwörter wählen oder ihre Kennwörter an exponierten Stellen aufschreiben. Für die Systemadministratoren bedeuten zusätzliche Anwendungen einen erhöhten Aufwand für die Verzeichnisverwaltung und die Bewältigung einer zunehmenden Anzahl an Help-Desk-Anrufen, um Kennwörter zurückzusetzen. Einige der Probleme, die durch zusätzliche Anwendungen verursacht werden, können durch die Single-Sign-on-Funktionalität gelöst werden. Die Single-Sign-on-Funktionalität ermöglicht Benutzern den Zugriff auf mehrere gesicherte Anwendungen durch eine einmalige Authentifizierung.

Anmerkung: "Gesichert" bezieht sich auf Anwendungen, für die sich Benutzer vor dem Zugriff auf die Funktionalität authentifizieren müssen.

Single-Sign-on wird für die unterstützten Anwendungsserver unterstützt; dabei können alternative Mechanismen zusammen mit dem Cúram-Anmeldemodul verwendet werden. Die Implementierung einer SSO-Lösung liegt in der Zuständigkeit der angepassten Implementierung. Es ist empfehlenswert, ein Tool eines anderen Anbieters zu verwenden, z. B. IBM Tivoli oder CA SiteMinder.

In diesem Kapitel werden die Eigenschaften des Anwendungsservers beschrieben, die die Verwendung einer SSO-Lösung ermöglichen.

8.2 Single Sign-on mit WebSphere

Wenn SSO für WebSphere erforderlich ist, können hierfür ein einfacher WebSphere-Mechanismus zur Authentifizierung über Dritte (LTPA) sowie zusätzliche angepasste Anmeldemodule verwendet werden. Das LTPA-Protokoll führt zu einem Token, das für einen authentifizierten Benutzer erstellt wird. In WebSphere wird ein Token generiert, sobald Berechtigungsnachweise für einen authentifizierten Benutzer hinzugefügt werden. Dieses Token wird dann zum Abrufen von Identitätsinformationen für einen authentifizierten Benutzer in einer SSO-Umgebung verwendet.

Sicherheit wird als Cúram-Anmeldemodul innerhalb einer Kette logischer Module implementiert, die in WebSphere eingerichtet wurden. Mindestens eines dieser Anmeldemodule sollte für das Hinzufügen von Berechtigungsnachweisen für den Benutzer zuständig sein. Das Cúram-Anmeldemodul fügt standardmäßig Berechtigungsnachweise für einen authentifizierten Benutzer hinzu. Daraus ergibt sich, dass die konfigurierte WebSphere-Benutzerregistry, die von einem nachfolgenden Anmeldemodul ausgeführt wird, keine Berechtigungsnachweise hinzufügt. Der empfohlene Ansatz für die Implementierung einer SSO-Lösung ist das Hinzufügen eines angepassten Anmeldemoduls an einer beliebigen Stelle in der Anmeldemodulkette.

Es besteht die Möglichkeit, das Hinzufügen von Berechtigungsnachweisen für einen nicht authentifizierten Benutzer zu inaktivieren und somit die Implementierung einer SSO-Lösung zu ermöglichen.

Das JAAS-Anmeldemodul von Cúram für WebSphere prüft, ob in WebSphere ein LTPA-Token existiert; hierfür wird der Callback "WSCredTokenCallbackImpl" für WebSphere verwendet. Wenn dieses Token existiert und gültig ist, führt das Cúram-Anmeldemodul keine Authentifizierung durch.

Es können Berechtigungsnachweise zur WebSphere-Benutzerregistry hinzugefügt werden. Berechtigungsnachweise umfassen Authentifizierungsinformationen zur Benutzeranmeldung, einschließlich der eindeu-

tigen ID für den Benutzer. WebSphere überprüft, ob für einen Benutzer Berechtigungsnachweise vorhanden sind, nachdem alle konfigurierten Systemanmeldemodule ausgeführt wurden; falls die Berechtigungsnachweise existieren, wird die WebSphere-Benutzerregistry nicht abgefragt. Berechtigungsnachweise werden vom JAAS-Anmeldemodul von Cúram nicht hinzugefügt, wenn die folgenden Einstellungen gelten:

- Eigenschaft "curam.security.check.identity.only" ist auf "true" gesetzt
- Eigenschaft "curam.security.user.registry.enabled" ist auf "true" gesetzt

Wie bereits in 7.4, „Bereitstellung einer externen Anwendung”, auf Seite 30 erwähnt, gibt es Eigenschaften, die sich auf den Typ des externen Benutzers beziehen und die steuern, ob Berechtigungsnachweise für einen bestimmten externen Benutzertyp zu WebSphere hinzugefügt werden. Zu diesen zählen:

- Eigenschaft "curam.security.user.registry.enabled.types"
- Eigenschaft "curam.security.user.registry.disabled.types"

Durch diese Eigenschaften ist eine differenziertere Steuerung der Authentifizierung für externe Benutzertypen möglich.

Falls das JAAS-Anmeldemodul von Cúram keine Berechtigungsnachweise hinzufügt, wird eine Abfrage an die WebSphere-Benutzerregistry abgesetzt, um zu versuchen, ob Berechtigungsnachweise für den Benutzer hinzugefügt werden können.

8.3 Single Sign-on für WebLogic Server

Wenn SSO für WebLogic Server erforderlich ist, kann der Authentifizierungsprovider von WebLogic Server oder ein angepasster Authentifizierungsprovider diese Funktion bereitstellen. Weitere Informationen zu Authentifizierungsprovidern finden Sie in der Dokumentation zu WebLogic Server. Für WebLogic Server sind Berechtigungsnachweise/Principals und die Gruppe erforderlich, zu der der Benutzer gehört; diese müssen vom konfigurierten Authentifizierungsprovider hinzugefügt werden. Für eine SSO-Lösung fügt das JAAS-Anmeldemodul von Cúram keine Berechtigungsnachweise zu dem JAAS-Subjekt hinzu, durch die es einem alternativen Authentifizierungsprovider möglich wäre, für das Hinzufügen von Berechtigungsnachweisen zuständig zu sein.

Berechtigungsnachweise werden nicht hinzugefügt, wenn die folgenden Einstellungen gelten:

- Eigenschaft "curam.security.check.identity.only" ist auf "true" gesetzt
- Eigenschaft "curam.security.user.registry.enabled" ist auf "true" gesetzt

Wie bereits in 7.4, „Bereitstellung einer externen Anwendung”, auf Seite 30 erwähnt, gibt es Eigenschaften, die sich auf den Typ des externem Benutzers beziehen und die steuern, ob Berechtigungsnachweise für einen bestimmten externen Benutzertyp zu WebLogic Server hinzugefügt werden. Zu diesen zählen:

- Eigenschaft "curam.security.user.registry.enabled.types"
- Eigenschaft "curam.security.user.registry.disabled.types"

Durch diese Eigenschaften ist eine differenziertere Steuerung der Authentifizierung für externe Benutzertypen möglich.

Die Zuständigkeit für das Hinzufügen von Berechtigungsnachweisen wird einem anderen Authentifizierungsprovider überlassen, d. h. dem Hauptauthentifizierungsprovider für die Authentifizierung des Benutzers. In einem SSO-Szenario muss während der Ausführung Methode commit() des Anmeldemoduls für einen Benutzer lediglich ein einziger Authentifizierungsprovider Berechtigungsnachweise zu dem JAAS-Subjekt hinzufügen.

Kapitel 9. Weitere Sicherheitsaspekte

9.1 Übersicht

Ein weiteres zentrales Sicherheitsproblem ist der Schutz des Inhalts bei dessen Eingabe, Anzeige und Übertragung im Netz an die IBM Cúram Social Program Management-Anwendung. Bei der Standardkonfiguration wird SSL verwendet; mit dem vom Anwendungsserver bereitgestellten SSL wird der Inhalt bei der Übertragung gesichert.

Zusätzlich werden während des Entwicklungszyklus branchenspezifische, führende Produkte verwendet, um Verstöße gegen die Sicherheit in der Anwendung regelmäßig zu überwachen. Beispiele für solche möglichen Verstöße sind siteübergreifendes Scripting und die SQL-Injection. Solche Sicherheitsrisiken werden, sobald sie aufgespürt werden, innerhalb der Infrastruktur aufgelöst.

Es ist empfehlenswert, dass Kunden ein ähnliche Sicherheitsüberwachung für ihre Anwendung durchführen.

9.2 SSL-Einstellungen für die Anwendung

Standardmäßig ist SSL für den Zugriff auf die Webanwendung aktiviert. Hierdurch wird eine gesicherte SSL-Verbindung zwischen dem Client und dem Server sichergestellt und die Verschlüsselung der Daten gewährleistet. SSL wird über die Einstellungen in der Datei `web.xml` für die Web-Client-Anwendung aktiviert. SSL wird auf der Anwendungsserverebene durch Einstellungen in WebLogic Server und WebSphere aktiviert. Diese Einstellungen für die Anwendungsserver werden über die IBM Cúram Social Program Management-Konfigurationsscripts vorgenommen.

Wichtiger Hinweis: Mit den Konfigurationsscripts wird gewährleistet, dass SSL standardmäßig aktiviert ist; dies ist jedoch eine Standardkonfiguration, die aktualisiert werden muss und für das SSL-Protokoll müssen neue Zertifikate eingerichtet werden.

Es ist empfehlenswert, SSL für den Zugriff auf die IBM Cúram Social Program Management-Anwendung aktiviert zu lassen; in Abhängigkeit von den jeweiligen Projektkonfigurationen kann es jedoch erforderlich sein, SSL für die Anwendung zu inaktivieren.

Es ist zwar möglich, SSL zu inaktivieren, aber nicht empfehlenswert. In 10.9, „Inaktivieren der SSL-Einstellungen für die Anwendung”, auf Seite 39 finden Sie weitere Details.

Kapitel 10. Anpassen der Authentifizierung

10.1 Anpassen der Anmeldeseite

Die Standardanmeldeanzeige wird durch die Datei `logon.jsp` im Verzeichnis `lib/curam/web/jsp` von Client Development Environment for Java (CDEJ) dargestellt. Die Datei `logon.jsp` kann durch Erstellen einer Kopie der Standarddatei und durch Platzieren dieser Datei im Ordner `webclient/components/<angepasst>/WebContent` angepasst werden; dabei ist *<angepasst>* der Name der angepassten Web-Client-Komponente.

Im Abschnitt zu den Anmeldeseiten im Referenzhandbuch zu *Cúram Web Client* sind Richtlinien dazu enthalten, welche Angaben in der Datei `logon.jsp` bestehen bleiben müssen; in diesem Abschnitt finden Sie auch weitere Details.

10.2 Anwenden der Darstellung auf die Anmeldeseite

Darstellungsänderungen können auf herkömmliche Weise auf die Anmeldeseite `logon.jsp` angewendet werden, d. h. durch Hinzufügen des relevanten CSS zu allen CSS-Dateien in der angepassten Komponente. Im Referenzhandbuch zu *Cúram Web Client* finden Sie Details zur Darstellung.

10.3 Aktivieren von Benutzernamen mit Erweiterungszeichen für WebLogic Server

Dieser Abschnitt kann ignoriert werden, wenn der WebLogic Server-Anwendungsserver nicht verwendet wird.

Falls Sie über Cúram-Benutzernamen oder -Kennwörter mit Erweiterungszeichen verfügen (z. B. "üßer"), stellt WebLogic Server das proprietäre Attribut **j_character_encoding** bereit, das zur formularbasierten Anmeldeseite **logon.jsp** hinzugefügt werden muss. Weitere Informationen finden Sie in der Dokumentation zu WebLogic Server. Das Attribut muss zum Tabellenelement in der Datei `logon.jsp` wie im Folgenden dargestellt hinzugefügt werden: 10.3, „Aktivieren von Benutzernamen mit Erweiterungszeichen für WebLogic Server”.

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

Abbildung 6. WebLogic Server-Unterstützung für Anmeldungen mit Erweiterungszeichen

10.4 Ändern der Groß-/Kleinschreibung des Benutzernamens

Mit der Eigenschaft "curam.security.casesensitive" wird die Groß-/Kleinschreibung von Benutzernamen gesteuert. Standardmäßig ist diese Eigenschaft in der Datei `Application.prx` auf `true` gesetzt. Wenn diese Eigenschaft in der Datei `Application.prx` auf `false` gesetzt ist, wird bei Authentifizierungs- und Berechtigungsmechanismen die Groß-/Kleinschreibung ignoriert.

Weitere Details zur Datei `Application.prx` finden Sie im Kapitel zu den Cúram-Konfigurationseinstellungen im Handbuch *Cúram Server Developer*.

10.5 Hinzufügen von angepassten Überprüfungen zum Authentifizierungsprozess

Um angepasste Überprüfungen hinzuzufügen, muss die Schnittstelle `curam.util.security.CustomAuthenticator` implementiert werden. Diese Schnittstelle weist eine Methode auf: `authenticateUser()`. Die Methode `authenticateUser()` wird sowohl für die Standardauthentifizierung als auch für die ausschließliche Authentifizierung nach Identität aufgerufen. Das Ergebnis dieser Methode soll ein Eintrag aus der Codetabelle `curam.util.codetable.SECURITYSTATUS` sein. Im Fall einer erfolgreichen Authentifizierung muss das Ergebnis `curam.util.codetable.SECURITYSTATUS.LOGIN` lauten.

Bei Authentifizierungsfehlern kann ein beliebiges Ergebnis, einschließlich null, zurückgegeben werden. Es ist allerdings empfehlenswert, einen anderen Code aus der Codetabelle `curam.util.codetable.SECURITYSTATUS` zu verwenden. Diese Codetabelle kann so erweitert werden, dass angepasste Codes wie im Kapitel zu den Codetabellen im Handbuch *Cúram Server Developer* beschrieben, einbezogen werden.

Nach dem Aufrufen der angepassten Überprüfungen werden durch den Authentifizierungsprozess die relevanten Felder in der Benutzerdatenbanktabelle aktualisiert. Wenn beispielsweise das Ergebnis der angepassten Überprüfungen nicht `SECURITYSTATUS.LOGIN` lautet, wird die Anzahl der Anmeldefehler um 1 erhöht; wenn der Schwellenwert für nicht autorisierten Zugriff erreicht wird, wird das Konto inaktiviert. Wenn das Ergebnis hingegen `SECURITYSTATUS.LOGIN` lautet, werden die Anmeldefehler auf 0 zurückgesetzt und das Feld für die letzte erfolgreiche Anmeldung wird aktualisiert.

Anmerkung: Ist die ausschließliche Authentifizierung nach Identität aktiviert, werden die Felder der Benutzerdatenbanktabelle nicht aktualisiert, unabhängig vom Ergebnis der angepassten Überprüfung.

10.6 Konfigurieren des angepassten Authentifikators

Für die Konfiguration der Anwendung zur Verwendung dieser angepassten Erweiterung muss die Eigenschaft "curam.custom.authentication.implementation" in der Datei `Application.prx` auf den vollständig qualifizierten Namen der Klasse gesetzt werden, die die Schnittstelle `CustomAuthenticator` implementiert.

Weitere Details zur Datei `Application.prx` finden Sie im Kapitel zu den Cúram-Konfigurationseinstellungen im Handbuch *Cúram Server Developer*.

10.7 Konfigurieren der ausschließlichen Authentifizierung nach Identität

Um die ausschließliche Authentifizierung nach Identität zu konfigurieren, muss die Eigenschaft "curam-.security.check.identity.only" in der Datei `AppServer.properties` auf `true` gesetzt werden, bevor das Konfigurationsziel (**`configure`**) ausgeführt wird. Es ist außerdem möglich, diese Eigenschaft zu setzen, sobald die Anwendung mithilfe der Anwendungsserverkonsole bereitgestellt wurde. Weitere Informationen zur Konfiguration des Anwendungsservers finden Sie in den Handbüchern *Cúram Server Deployment*.

10.8 Hinzufügen der Callback-Schnittstelle für fehlerhafte Cacheaktualisierung

Die neue Callback-Klasse muss die Schnittstelle `curam.util.security.SecurityCacheFailureCallback` in eine Klasse implementieren, die einen öffentlichen Standardkonstruktor aufweist. Die Implementierung des Callback wird durch Setzen der Anwendungseigenschaft "curam.security.cache.failure.callback" auf den Namen der Implementierungsklasse registriert. Wenn die Eigenschaft nicht gesetzt ist, wird kein Versuch unternommen, einen Callback-Handler aufzurufen.

10.9 Inaktivieren der SSL-Einstellungen für die Anwendung

Standardmäßig ist SSL für den Zugriff auf die IBM Cúram Social Program Management-Anwendung aktiviert. Hierdurch wird eine gesicherte SSL-Verbindung zwischen dem Client und dem Server sichergestellt und die Verschlüsselung der Daten gewährleistet. SSL kann über die Einstellungen in der Datei `web.xml` für die Web-Client-Anwendung und auf der Anwendungsserverebene durch Einstellungen in WebLogic Server und WebSphere für den Client aktiviert und inaktiviert werden. Diese Einstellungen für die Anwendungsserver werden mittels Konfigurationsscripts konfiguriert. Es ist empfehlenswert, SSL für den Zugriff auf die Anwendung aktiviert zu lassen; in Abhängigkeit von den jeweiligen Projektkonfigurationen kann es jedoch erforderlich sein, SSL für die Anwendung zu inaktivieren. In den folgenden Abschnitten wird ausführlich beschrieben, wie dies funktioniert.

10.10 Modifizieren der Datei "web.xml" für die Clientanwendung

Eine Modifizierung kann erreicht werden, indem das Element <transport-guarantee> in der Datei `web.xml` von `CONFIDENTIAL` in `NONE` geändert wird. Beachten Sie, dass dadurch nicht der Zugriff auf den Web-Client über HTTPS inaktiviert wird, sondern zusätzlicher Zugriff über HTTP möglich wird. Weitere Details zur Modifizierung der Datei `web.xml` finden Sie im Abschnitt zur Anpassung des Webanwendungsdeskriptors im Referenzhandbuch zu *Cúram Web Client*. Im Folgenden sehen Sie ein Beispiel dafür, wie diese Eigenschaft festgelegt werden kann:

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

10.11 Modifizieren der Konfiguration des Anwendungsservers

Für die Modifizierung der Konfiguration für WebSphere gibt es zwei Möglichkeiten. Es wird der unten beschriebene erste Ansatz empfohlen.

- Verwenden Sie den vorhandenen nicht sicheren Port, der standardmäßig für Web-Services eingerichtet wird (empfohlene Methode). Dies gilt sowohl für SSL- als auch für Nicht-SSL-Verbindungen.
  1. Navigieren Sie zu "Umgebung -> Virtuelle Hosts -> Client-Host -> Hostaliasse
  2. Klicken Sie auf "Neu" und geben Sie * für den Hostnamen und 9082 für die Portnummer ein; klicken Sie anschließend auf OK.
  3. Klicken Sie auf der nächsten Seite auf "Speichern", um Ihren neuen Wert für die Serverkonfiguration zu speichern. Beachten Sie, dass der Port 9082 der in der Standardclientanwendung konfigurierten Web-Services-Kette von Curam (*CuramWebServicesChain*) entspricht; dieser Port ist nun der Port, der für den Zugriff auf die Anwendung mit HTTP verwendet werden kann.
- Verwenden Sie den aktuellen SSL-Port 9044 wieder:

  Der aktuelle Port kann als nicht-sicherer Port eingerichtet werden. Die hierfür auszuführenden Schritte werden im Handbuch *Cúram Deployment for WebSphere Application Server* in Abschnitt A.2.11 zur Serverkonfiguration bzw. zur Einrichtung des Portzugriffs beschrieben. Führen Sie die Schritte 7 bis einschließlich 11 aus. Der einzige Unterschied betreffend Schritt 11 ist, dass die Vorlage für die Transportkette auf "WebContainer" gesetzt werden muss (und nicht auf "WebContainer Secure").

10.12 Analysieren der Datenbanktabelle "AuthenticationLog"

Alle Authentifizierungsversuche (erfolgreiche und fehlgeschlagene Versuche) werden in der Datenbanktabelle "AuthenticationLog" protokolliert. Die folgenden Zeilen dieser Tabelle sind die aussagekäftigsten:

Tabelle 1. Inhalt des Authentifizierungsprotokolls

Feld	Bedeutung
timeEntered	Zeitmarke des Protokolleintrags.
userName	Benutzername, der für den Anmeldeversuch verwendet wurde.

Tabelle 1. Inhalt des Authentifizierungsprotokolls (Forts.)

Feld	Bedeutung
altLogin	Boolesche Information, ob der Benutzername eine alternative Anmelde-ID darstellt. Wenn diese Spalte dem Wert "1" (true) entspricht, ist der Wert in der Spalte "userName" eine alternative Anmelde-ID (gemäß 2.5, „Alternative Anmelde-IDs", auf Seite 5). Anderenfalls stellt die Spalte "userName" den Benutzernamen aus der Tabelle "Users" oder "ExternalUser" dar.
loginFailures	Anzahl der Anmeldefehler für diesen Benutzer seit der letzten erfolgreichen Anmeldung.
lastLogin	Datum und Uhrzeit der letzten erfolgreichen Anmeldung.
loginStatus	Status des Anmeldeversuchs. Es gibt die folgenden Status: • LOGIN: Erfolgreiche Anmeldung. • ACCDISABLE: Das Konto wurde explizit inaktiviert. • ACCEXPIRED: Das Ablaufdatum für das Kennwort wurde erreicht. • PWDEXPIRED: Die Anzahl der Tage, während denen der Benutzer sein Kennwort ändern kann, wurde überschritten. • BADUSER: Der Benutzer existiert nicht. • AUTHONLY: Wird für eine ausschließliche Authentifizierung nach Identität verwendet und gibt an, dass nur Berechtigungsüberprüfungen durchgeführt werden. • BADPWD: Das angegebene Kennwort war falsch. • BREAKIN: Es wurde eine angegebene Anzahl von falschen Kennwörtern erreicht. Das Konto wird inaktiviert. • RESTRICTED: Der Benutzer darf momentan nicht auf das System zugreifen. • LOGEXPR: Die Anzahl der Anmeldeversuche, während denen der Benutzer sein Kennwort ändern kann, wurde überschritten. • AMBIGUOUS: Der angegebene Benutzername ist mehrdeutig; er ist ein Duplikat eines anderen Benutzernamens ohne Berücksichtigung der Groß-/Kleinschreibung.

Die API `LogAdmin` kann für die Abfrage der Datenbanktabelle "AuthenticationLog" verwendet werden. Weitere Details finden Sie in der Java-Dokumentation für diese Klasse.

Kapitel 11. Anpassen der Berechtigung

11.1 Übersicht

In diesem Kapitel wird die Einrichtung der Benutzerberechtigung ausführlich beschrieben.

11.2 Erstellen der Berechtigungsdatenzuordnung

Die Berechtigungsdaten für einen Benutzer können mithilfe der DMX-Dateien (Data Manager) oder über die Cúram-Administrationsanzeigen eingerichtet werden. Details zur Feststellung, wie Sicherheit über eine Geschäftsperspektive organisiert wird, finden Sie im Handbuch *Cúram System Configuration*.

Für die Erstellung einer neuen Sicherheitsrolle für einen Benutzer müssen die Sicherheits-IDs (SIDs), auf die der Benutzer zugreifen können muss, angegeben werden. Diese SIDs müssen dann in SID-Gruppen angeordnet werden. Die Rolle, die Gruppen sowie die SIDs müssen, sobald sie angegeben sind, für die Sicherheitstabellen eingerichtet werden, die sie darstellen.

Sicherheitsdaten sind für die Einrichtung einer IBM Cúram Social Program Management-Anwendung von zentraler Bedeutung. Die unten stehenden Beispiele beschreiben das Hinzufügen von Sicherheitsdaten zum Verzeichnis `data/initial` innerhalb der Komponente.

11.3 Erstellen einer neuen Sicherheitsrolle

Für die Erstellung einer neuen Sicherheitsrolle muss ein Eintrag zu der Datenbanktabelle "SecurityRole" hinzugefügt werden, mit dem das Attribut `rolename` definiert wird.

Hierfür müssen Sie die Datei `SecurityRole.dmx` im Verzeichnis `%SERVER_DIR%/components/<angepasst>/data/initial` erstellen oder einen Eintrag zu dieser Datei hinzufügen; dabei ist `<angepasst>` jedes unter "components" neu erstellte Verzeichnis, das derselben Verzeichnisstruktur entspricht wie `components/core`.

11.4 Erstellen einer neuen Sicherheitsgruppe

Für die Erstellung einer neuen Sicherheitsgruppe muss ein Eintrag zu der Datenbanktabelle "SecurityGroup" hinzugefügt werden, mit dem das Attribut `groupname` definiert wird.

Hierfür müssen Sie die Datei `SecurityGroup.dmx` im Verzeichnis `%SERVER_DIR%/components/<angepasst>/data/initial` erstellen oder einen Eintrag zu dieser Datei hinzufügen; dabei ist `<angepasst>` jedes unter "components" neu erstellte Verzeichnis, das derselben Verzeichnisstruktur entspricht wie `components/core`.

11.5 Verknüpfen der Sicherheitsgruppe mit der Sicherheitsrolle

Die Sicherheitsrolle und die Sicherheitsgruppe müssen miteinander verknüpft werden. Erstellen Sie hierfür in der Tabelle "SecurityRoleGroup" einen neuen Eintrag, wodurch die Attribute `rolename` und `groupname` definiert werden.

Hierfür müssen Sie die Datei `SecurityRoleGroup.dmx` im Verzeichnis `%SERVER_DIR%/components/<angepasst>/data/initial` erstellen oder einen Eintrag zu dieser Datei hinzufügen; dabei ist `<angepasst>` jedes unter "components" neu erstellte Verzeichnis, das derselben Verzeichnisstruktur entspricht wie `components/core`.

11.6 Erstellen der Sicherheits-ID (SID)

Für die Erstellung einer neuen SID muss ein Eintrag zu der Tabelle "SecurityIdentifier" hinzugefügt werden, wodurch die Attribute sidname und sidtype definiert werden.

Hierfür müssen Sie die Datei SecurityIdentifier.dmx im Verzeichnis %SERVER_DIR%/components/<angepasst>/data/initial erstellen oder einen Eintrag zu dieser Datei hinzufügen; dabei ist <angepasst> jedes unter "components" neu erstellte Verzeichnis, das derselben Verzeichnisstruktur entspricht wie components/core.

11.7 Verknüpfen der Sicherheitsgruppe mit der SID

Für das Verknüpfen der Sicherheitsgruppe mit der SID muss ein Eintrag zu der Tabelle "SecurityGroup-SID" hinzugefügt werden, wodurch die Attribute groupname und sidname definiert werden.

Hierfür müssen Sie die Datei SecurityGroupSID.dmx im Verzeichnis %SERVER_DIR%/components/<angepasst>/data/initial erstellen oder einen Eintrag zu dieser Datei hinzufügen; dabei ist <angepasst> jedes unter "components" neu erstellte Verzeichnis, das derselben Verzeichnisstruktur entspricht wie components/core.

11.8 Verknüpfen der Sicherheitsrolle mit dem Benutzer

Um Berechtigungsdaten einem Benutzer zuzuordnen, muss die Sicherheitsrolle mit dem Benutzer verknüpft werden.

Hierfür müssen Sie den Eintrag für den angegebenen Benutzer in der Datei Users.dmx im Verzeichnis %SERVER_DIR%/components/<angepasst>/data/initial aktualisieren; dabei ist <angepasst> jedes unter "components" neu erstellte Verzeichnis, das derselben Verzeichnisstruktur entspricht wie components/core; dabei wird das Attribut rolename für den Rollennamen verwendet, der in der Tabelle "SecurityRole" angegeben ist.

11.9 Laden von Sicherheitsinformationen in die Datenbank

Sobald alle Informationen in den unterschiedlichen DMX-Dateien eingegeben wurden, muss der Data Manager zum Laden der DMX-Daten in die Datenbank verwendet werden. Weitere Details finden Sie im Kapitel zum Data Manager im Handbuch *Cúram Server Developer*.

11.10 Erstellen von Funktions-IDs (FIDs)

Wenn auf eine Methode öffentlich zugegriffen werden kann (indem das Stereotyp auf <<facade>> gesetzt wird), wird die Sicherheit automatisch eingeschaltet. Dies bedeutet, dass für diese Methode automatisch eine SID generiert wird und das Flag für die aktivierte Sicherheit für die Methode ist auf true gesetzt. Die SID und ihr Flag fidenabled werden in der datenbankunabhängigen Datei <Projektname>_Fids.xml im Unterverzeichnis /build/svr/gen/ddl gespeichert. Diese Datei wird zum Einfügen der FID-Informationen in die Datenbank über den Data Manager verwendet.

Für eine FID ist die folgende Namenskonvention zu beachten: <Klassenname>.<Methodenname>; eine FID darf höchstens aus 100 Zeichen bestehen. Beispiel: Für ein Geschäftsprozessobjekt mit dem Namen ProductEligibility und den beiden Methoden insertProduct und testProduct werden zwei FIDs erstellt: ProductEligibility.insertProduct und ProductEligibility.testProduct.

Wenn die Sicherheitsfunktion im Modell für eine Prozessmethode zur Entwurfszeit inaktiviert wird, wird dennoch eine SID/FID generiert; das Flag für die aktivierte Sicherheit ist jedoch auf false gesetzt. Das Setzen des Flags für die aktivierte Sicherheit auf false bedeutet, dass für diese Methode keine Berechtigungsprüfung durchgeführt wird.

11.11 Ausschalten der Sicherheit für eine Prozessmethode

Durch das Setzen der Option `Generate_Security` für die Prozessmethode auf `false` in dem Modell wird die Sicherheit für eine Prozessmethode ausgeschaltet.

Wenn die Sicherheitsfunktion für eine Prozessmethode zur Entwurfszeit im Modell inaktiviert wird, wird dennoch eine Funktions-ID generiert; das Flag für die aktivierte Sicherheit ist jedoch auf `false` gesetzt. Das Setzen des Flags für die aktivierte Sicherheit auf "false" bedeutet, dass für diese Methode keine Berechtigungsprüfung durchgeführt wird.

11.12 Sicherheitsaspekte während der Entwicklung

Es ist wichtig, die Auswirkungen dieser Entwurfsoptionen bei der Sicherheitsimplementierung während der Entwicklung einer IBM Cúram Social Program Management-Anwendung zu berücksichtigen. Sie sind die Instanzen zur Abwehr von unbefugtem Zugriff auf die Anwendungsprozessfunktion. Allgemein gesagt, wird die Sicherheit für die meisten Prozessmethoden eingeschaltet. Die Sicherheit kann für eine Prozessmethode ausgeschaltet werden, für die keine Sicherheit erforderlich ist, z. B. für eine Anmeldemethode, die aufgerufen wird, wenn ein Benutzer versucht, sich bei einer Anwendung anzumelden. Wenn ein Benutzer noch nicht authentifiziert oder berechtigt wurde, benötigt er für die Anmeldung Zugriff auf diese Methode; aus diesem Grund kann es erforderlich sein, die Sicherheit für diese Methode auszuschalten.

Während der Anfangsentwurfsphase einer Anwendung kann der Aufwand, der erforderlich ist, um die Sicherheitsumgebung mit einer zukünftigen Anwendung „synchron" zu halten, sehr groß sein. Es ist möglich, die Berechtigungsprüfung zu inaktivieren; hierfür muss die Eigenschaft "curam.security.disable.authorisation" in der Datei "Application.prx" festgelegt werden.

Warnung: Warnung

Die Eigenschaft "curam.security.disable.authorisation" sollte nur für die Entwurfsphase aktiviert werden. In einer Produktionsumgebung sollte sie niemals auf `true` gesetzt werden.

Schließlich sollte darauf hingewiesen werden, dass die einer Informationen, denen eine FID zugeordnet ist, nicht geändert werden können, sobald der Code und die Scripts aus einem Arbeitsmodell generiert wurden. Um diese Informationen zu ändern, ist eine Änderung des Modells und eine erneute Generierung und Erstellung der Datenbank erforderlich.

11.13 Steuern der Protokollierung von Berechtigungsfehlern für den Client

Standardmäßig werden Berechtigungsfehler für den Web-Client nicht aufgezeichnet.

Mit der Eigenschaft "curam.enable.logging.client.authcheck" wird gesteuert, ob die für den Web-Client aufgetretenen Berechtigungsfehler protokolliert werden oder nicht. Standardmäßig ist diese Eigenschaft auf `false` gesetzt; dies bedeutet, dass die Fehler nicht protokolliert werden. Wenn diese Eigenschaft auf `true` gesetzt ist, wird ein Protokoll dieser Berechtigungsfehler in der Datenbanktabelle "AuthorisationLog" gespeichert. Weitere Informationen zu dieser Eigenschaft finden Sie im Abschnitt zu den dynamischen Eigenschaften der Datei "Application.prx" im Handbuch *Cúram Server Developer*.

11.14 Berechtigen neuer SID-Typen

Es wird eine Serverschnittstellenmethode bereitgestellt, mit der die Berechtigung direkt ausgeführt werden kann. Diese Methode kann zu einer Klasse hinzugefügt werden, die Daten des konzeptionellen Elements bearbeitet, das mit dem neuen SID-Typ gesichert wird.

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

Im Folgenden sehen Sie ein Nutzungsbeispiel für diese Methode:

```
// Die dem zu sichernden konzeptionellen Element
       // zugeordnete SID.
       String someSID = "beliebigeSID";

       // Abrufen des angemeldeten Benutzernamens
       String loggedUser =
         curam.util.transaction.TransactionInfo.getProgramUser();

       // Überprüfen, ob der Benutzer über Zugriffsrechte verfügt
       if (curam.util.security.Authorisation.isSIDAuthorised(
             someSID, loggedUser)) {
         // Durchführen einer vertraulichen Aktion, für die dieser Benutzer über Rechte verfügt
         ...
       } else {
       // Ausgeben einer Ausnahmebedingung, die darauf hinweist, dass der Benutzer über keine Zugriffsrechte
       // zur Durchführung dieser Aktion verfügt
         AppException exception
           = new AppException(MESSAGE.ERR_USER_NO_ACCESS);
         throw exception;
       }
```

Abbildung 7. Beispielsyntax für isSIDAuthorised()

11.15 Analysieren der Datenbanktabelle "AuthorisationLog"

Alle Berechtigungsfehler werden in der Datenbanktabelle "AuthenticationLog" protokolliert. Die folgenden Zeilen dieser Tabelle sind die aussagekäftigsten:

Tabelle 2. Inhalt des Authentifizierungsprotokolls

Feld	**Bedeutung**
timeEntered	Zeitmarke des Protokolleintrags.
userName	Der Benutzername zum Berechtigungsversuch.
identifierName	Die Sicherheits-ID (SID) oder Funktions-ID (FID) zu diesem Fehler.

Die API `LogAdmin` kann für die Abfrage der Datenbanktabelle "AuthorisationLog" verwendet werden. Weitere Details finden Sie in der Java-Dokumentation für diese Klasse.

Kapitel 12. Anpassen der Verschlüsselung

12.1 Übersicht

In diesem Kapitel wird die Konfiguration und Anpassung der Verschlüsselung in IBM Cúram Social Program Management beschrieben.

12.2 Chiffrierungsanpassung

Die Modifikation der Standard-Chiffrierungseigenschaften ist relativ einfach, muss jedoch genau geplant und getestet werden. Es ist ein Neustart erforderlich, damit die Änderungen implementiert werden. Je nach Größe und Topologie Ihres Unternehmens und Ihrer Bereitstellungen müssen Sie einen Zeitpunkt wählen, zu dem getätigte Änderungen keine negativen Auswirkungen haben. Berücksichtigen Sie außerdem, dass alle Daten (z. B. Eigenschaften mit verschlüsselten Kennwörtern), die von Cúram Transport Manager (CTM) verwaltet werden, entweder aktualisiert oder verwaltet werden müssen, damit die Systeme untereinander synchron bleiben. (Weitere Informationen dazu finden Sie im Handbuch *Cúram Transport Manager Guide*.)

Die Modifikation der Standard-Chiffrierungseigenschaften umfasst folgende Schritte:

1. Wählen Sie neue Einstellungen für `CryptoConfig.properties` und zugrunde liegende Artefakte aus (siehe 4.5, „Cúram-Chiffrierungseigenschaften", auf Seite 20).
2. Führen Sie je nach Einstellungen zusätzliche Schritte durch (z. B. Modifizieren des Keystore gemäß 12.4, „Vorgehensweise beim Erstellen eines neuen Keystore", auf Seite 46).
3. Modifizieren Sie die Datei `CryptoConfig.properties`. Beachten Sie, dass der Standardspeicherort `<SERVER_DIR>/project/properties` lautet.
4. Entfernen Sie vorhandene `CryptoConfig.jar`-Dateien (enthalten `CryptoConfig.properties`), die sich im Verzeichnis `<JAVA_HOME>/jre/lib/ext` (`$JAVA_HOME/lib/ext` unter IBM z/OS) befinden. Wenn Cúram-Clients oder -Server ausgeführt werden, müssen diese beendet werden, damit die aktualisierte Datei `CryptoConfig.jar` mit den aktualisierten Einstellungen implementiert werden kann.
5. Verschlüsseln Sie die Kennwörter in allen vorhandenen Eigenschaftendateien erneut, die gemäß 4.7, „Per Chiffrierung verschlüsselte Kennwörter", auf Seite 22 identifiziert wurden. Die Apache Ant-Ziele "configtest", "configure" und "installapp" platzieren die aktualisierte Datei `CryptoConfig.jar` in das Java-Verzeichnis `lib/ext`.
6. Testen und überprüfen Sie Ihre Änderungen.

Das Testen Ihrer Änderungen umfasst das Überprüfen von Funktionen, die beeinträchtigt werden können. Beispiele:

- Stellen Sie sicher, dass das Ant-Ziel "configtest" weiterhin funktioniert.
- Stellen Sie sicher, dass die Stapelprogramme weiterhin funktionieren.
- Stellen Sie gegebenenfalls sicher, dass das Ant-Ziel "configure" weiterhin funktioniert.

Zugehörige Themen:

- 4.6, „Cúram-Digest-Eigenschaften", auf Seite 21
- 4.7, „Per Chiffrierung verschlüsselte Kennwörter", auf Seite 22

12.3 Schlüsselmanagement

Das Management geheimer Schlüssel für verschlüsselte Cúram-Kennwörter erfolgt über den von JDK bereitgestellten Befehl **keytool** oder über einen ähnlichen Befehl. Sie müssen lokale Entscheidungen über die Platzierung und Isolation des geheimen Schlüssels für Cúram treffen, die Ihrem lokalen Unternehmen und den Standards entsprechen.

Hinweis: Einige Einstellungen, die dem Befehl **keytool** übergeben wurden, müssen in den `CryptoConfig.properties`-Einstellungen berücksichtigt werden, die für eine erfolgreiche Bereitstellung koordiniert werden müssen (siehe 12.2, „Chiffrierungsanpassung", auf Seite 45). In der folgenden Tabelle wird die Beziehung zwischen den **keytool**-Befehlsargumenten und den Cúram-Verschlüsselungseigenschaften dargestellt.

Tabelle 3. Beziehung der "keytool"-Befehlsargumente zu den Cúram-Verschlüsselungseigenschaften

"keytool"-Argument	Eigenschaft "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

Anmerkung: Das geheime Schlüsselkennwort ist standardmäßig das Kennwort für den Schlüsselspeicher und darf nicht geändert werden.

Weitere Informationen zur Verwendung des Befehls **keytool** finden Sie in der JDK-Dokumentation.

Zugehörige Themen:

- 4.5, „Cúram-Chiffrierungseigenschaften", auf Seite 20
- 4.4, „Verschlüsselungseigenschaften", auf Seite 20
- 12.4, „Vorgehensweise beim Erstellen eines neuen Keystore"

12.4 Vorgehensweise beim Erstellen eines neuen Keystore

Für das Erstellen eines neuen Keystore, um den Cúram-Standardwert zu ersetzen, erfordert Folgendes: das Ausführen des Befehls **keytool** aus dem JDK (oder ähnliches), das Bearbeiten der Einstellungen "CryptoConfig.properties" (nur notwendig, wenn der Keystore-Name bzw. -Speicherort vom Standardnamen/-speicherort abweicht; das Ändern des Namens verdeutlicht jedoch Ihre Anpassungen) sowie die Sicherstellung, dass die Cúram-Ant-Ziele den neuen Keystore finden (nur notwendig, wenn der Standardspeicherort geändert wird).

Beispiel:

```
keytool -genseckey -v  -alias MySecretKey -keyalg AES -keysize 128 -keystore MyOrganization.keystore
-storepass secretpw -storetype jceks
```

Im Abschnitt 12.3, „Schlüsselmanagement" werden die **keytool**-Befehlsargumente in Bezug auf die Einstellungen `CryptoConfig.properties` identifiziert.

Der Standardspeicherort der Keystore-Datei ist das Verzeichnis `<SERVER_DIR>/project/properties` mit einer Unterverzeichnisstruktur, die das verwendete JDK widerspiegelt: „ibm" für das JDK von IBM und „sun" für das JDK von Oracle. Aus diesem Grund erwartet der Cúram-Build beim Erstellen einer Keystore-Datei, dass sie sich beim JDK von IBM in `<SERVER_DIR>/project/properties/ibm` befindet. Wenn Sie einen anderen Speicherort anstelle des Standardspeicherorts verwenden möchten, haben Sie die Wahl zwischen zwei Optionen:

1. Verwenden Sie einen absoluten Speicherort für die Keystore-Datei, wie in 4.4, „Verschlüsselungseigenschaften", auf Seite 20 beschrieben. In diesem Fall werden die Cúram-Standard-Keystore-Dateien in `CryptoConfig.jar` ignoriert und die absolute Einstellung `CryptoConfig.properties` bevorzugt verwendet.
2. Verwenden Sie die Ant-Eigenschaft `crypto.prop.file.location`, wenn Sie eines der Ziele ausführen, wie in 12.2, „Chiffrierungsanpassung", auf Seite 45 beschrieben, das `CryptoConfig.jar` erstellt und kopiert, um auf Ihren alternativen Speicherort hinzuweisen. Der angegebene Speicherort reflektiert die Struktur Ihres JDK - „ibm" oder „sun". Beispiel:
   - Platzieren Sie die neue Keystore-Datei unter Windows für das JDK von IBM in einen ähnlichen Speicherort wie diesen: `C:\Curam\keystore\ibm\MyOrganization.keystore`
   - Verweisen Sie zu diesem Speicherort, wenn die Build-Ziele ausgeführt werden: `ant configure -Dcrypto.prop.file.location=C:\Curam\keystore`

Anmerkung: Im oben genannten Beispiel ist für die Änderung der Keystore-Datei in `MyOrganization.keystore` eine entsprechende Änderung in `CryptoConfig.properties` erforderlich (siehe4.4, „Verschlüsselungseigenschaften", auf Seite 20).

Nach der Keystore-Erstellung müssen die in 12.2, „Chiffrierungsanpassung", auf Seite 45 beschriebenen Schritte durchgeführt werden.

Zugehörige Themen:

- 12.3, „Schlüsselmanagement", auf Seite 46
- 12.2, „Chiffrierungsanpassung", auf Seite 45

12.5 Digest-Anpassung

Die Modifikation der Standard-Digest-Einstellungen ist relativ einfach, muss jedoch genau geplant und getestet werden. Es ist ein Neustart erforderlich, damit die Änderungen implementiert werden. Je nach Größe und Topologie Ihres Unternehmens und Ihrer Bereitstellungen müssen Sie einen Zeitpunkt wählen, zu dem getätigte Änderungen keine negativen Auswirkungen haben. Berücksichtigen Sie außerdem, dass alle Daten (z. B. Benutzerkennwörter), die von Cúram Transport Manager (CTM) verwaltet werden, entweder aktualisiert oder verwaltet werden müssen, damit die Systeme untereinander synchron bleiben. (Weitere Informationen dazu finden Sie im Handbuch *Cúram Transport Manager Guide*.)

Dieser Prozess wird in 12.7, „Vorgehensweise beim Verwenden der ersetzten Digest-Einstellungen für einen Migrationszeitraum", auf Seite 48 ausführlich beschrieben.

Zugehörige Themen:

- 4.6, „Cúram-Digest-Eigenschaften", auf Seite 21
- 12.6, „Vorgehensweise beim Angeben eines Digest Salt"

12.6 Vorgehensweise beim Angeben eines Digest Salt

Cúram gibt keinen sofort einsatzfähigen Salt an, aber die Angabe eines Salts für mittels Digest verschlüsselte Kennwörter bietet einen Extraschutz gegen Brute-Force-Angriffe.

Geben Sie beim Angeben eines Salts für Ihre mittels Digest verschlüsselten Kennwörter wie folgt vor:

1. Wählen Sie eine ausreichend lange, zufällige Zeichenfolge aus.
2. Verschlüsseln Sie diese Zeichfolge mit dem Ant-Ziel **`encrypt`** (wie im Handbuch *Cúram Server Developer* beschrieben).
3. Platzieren Sie die verschlüsselte Zeichenfolge in eine Datei.

4. Geben Sie den Speicherort dieser Datei, die die verschlüsselte Salt-Zeichenfolge enthält, mit der Eigenschaft curam.security.crypto.digest.salt.location in CryptoConfig.properties an und stellen Sie sicher, dass alle implementierten CryptoConfig.jar-Dateien den aktualisierten Einstellungen entsprechen.

Aus Gründen der Verwaltbarkeit sollten Sie diese Änderungen in Verbindung mit den Schritten in 12.7, „Vorgehensweise beim Verwenden der ersetzten Digest-Einstellungen für einen Migrationszeitraum" durchführen.

12.7 Vorgehensweise beim Verwenden der ersetzten Digest-Einstellungen für einen Migrationszeitraum

Das Verwenden der ersetzten Digest-Einstellungen bedeutet, dass Sie Ihre vorhandenen mittels Digest verschlüsselten Kennwörter auf eine neue Verschlüsselungskonfiguration (z. B. neuer Salt) migrieren und Ihre Cúram-Benutzerkennwörter für einen Zeitraum lang automatisch migriert werden sollen. Dies gilt für interne und externe Cúram-Benutzer, allerdings nicht für Benutzer, die durch Drittanbieter-Sicherheitssysteme wie LDAP verwaltet werden.

Der Prozess lautet wie folgt:

1. Wählen Sie einen Zeitraum aus, in dem Ihr Cúram-System inaktiv sein kann und nicht ausgeführt wird.
2. Kopieren Sie die vorhandenen Digest-Eigenschaftsnamen und Werte aus CryptoConfig.properties und geben Sie den Eigenschaften die neuen ersetzten Eigenschaftsnamen.
3. Modifizieren Sie die vorhandenen Digest-Eigenschaftsnamen in CryptoConfig.properties.
4. Legen Sie die Eigenschaft curam.security.convertsupersededpassworddigests.enabled auf "true" fest.
5. Legen Sie die Eigenschaft curam.security.crypto.upgrade.start fest, um nachverfolgen zu können, wann Sie die aktualisierte Konfiguration eingeführt haben. Dieser Wert kann wie unten dargestellt verwendet werden, um nicht migrierte Benutzerkennwörter zu verwalten.
6. Starten Sie den Anwendungsserver neu, aber beachten Sie dabei Folgendes:

Anmerkung: Der Cúram-Standard-Web-Services-Benutzer (WEBSVCS) oder jeder Benutzer, der nicht über CuramLoginModule verarbeitet wird, steht für die automatische Kennwortmigration nicht zur Verfügung. Sie müssen diese Benutzer zurücksetzen, bevor Sie den Anwendungsserver neu starten. Dabei gehen Sie folgendermaßen vor:

1. Rufen Sie den neuen Digest-Kennwortwert über das Ant-Digest-Ziel (z. B. ant digest -Dpassword=password) ab.
2. Aktualisieren Sie den Kennwortwert in der Datenbank. Dies können Sie ganz einfach mit einer SQL-Anweisung durchführen (z. B. UPDATE USERS SET PASSWORD="<new digest value>' WHERE USERNAME='WEBSVCS';).
3. Jetzt können Sie den Anwendungsserver neu starten.

Legen Sie nach Ablauf der Migrationsphase (beispielsweise nach einigen Wochen oder Monaten) die Eigenschaft curam.security.convertsupersededpassworddigests.enabled auf "false" fest und deaktivieren Sie curam.security.crypto.upgrade.start property.

Benutzer, die sich während der Migrationsphase nicht angemeldet haben, können sich nicht mehr anmelden, da die Kennwörter nicht mehr übereinstimmen. Sie haben zwei Möglichkeiten, das Problem nicht aktualisierter Kennwörter zu lösen:

1. Fordern Sie diese Benutzer auf, Kontakt mit dem internen Support aufzunehmen, um ihre Kennwörter über die Admin-Benutzer-Schnittstelle zurücksetzen zu lassen.
2. Identifizieren Sie manuell in der Cúram-Tabelle "USERS" die Benutzer, die während der Migrationsphase aktualisiert wurden, und legen Sie ein neues Standardkennwort entweder über SQL (siehe **digest**-Ziel wie im Handbuch *Cúram Server Developer* zum Abrufen neuer Digest-Kennwortwerte be-

schrieben) oder über die Admin-Benutzer-Bildschirme manuell fest. Verwenden Sie beispielsweise die folgende Abfrage: `SELECT username FROM users WHERE lastwritten between timestamp('2013-06-01 15:00:00') AND timestamp('2013-09-01 00:00:00')`

Die Eigenschaft `curam.security.convertsupersededpassworddigests.enabled` darf aus folgenden Gründen nicht unbegrenzt auf "true" festgelegt werden:

1. Es ergibt wenig Sinn, die Konfiguration "A" auf Konfiguration "B" zu aktualisieren, während die ursprüngliche Konfiguration "A" aktiv bleibt;
2. Potenziell schwächere Verschlüsselungseigenschaften bleiben im System aktiv; und
3. Um diese Funktion für zukünftige Aktualisierungen verwenden zu können (z. B. von Konfiguration "B" zu "C"), müssen alle Kennwörter "A" mindestens auf "B" aktualisiert worden sein.

Anmerkung: Alle Dateien, z. B. DMX, mit gespeicherten Digests müssen in Hinblick auf Ihre Migrationsstrategie berücksichtigt werden, sodass die korrekten Werte wiedergegeben werden.

Anmerkung: Die Verwendung von Cúram Transport Manager (CTM) während der Migration muss in Hinblick auf die Sicherstellung kompatibler Einstellungen und Erwartungen zwischen den Quell- und Zielsystemen berücksichtigt werden.

Zugehörige Themen:

- 4.5, „Cúram-Chiffrierungseigenschaften”, auf Seite 20
- 4.6, „Cúram-Digest-Eigenschaften”, auf Seite 21

12.8 Modifizieren Ihrer Verschlüsselungskonfiguration für ein Produktionssystem

Die sofort einsatzfähigen OOTB-Verschlüsselungseigenschaften eignen sich ideal für typische Entwicklungs- oder Testumgebungen. Sie sollten aber für Produktionsumgebungen modifiziert werden, um diese Umgebungen zu schützen und die relativ risikoarmen Umgebungen von den risikoreichen Produktionsumgebungen zu isolieren.

Zu den typischen Änderungen der OTTB-Verschlüsselungseigenschaften gehören zur Vorbereitung der Produktion unter anderem folgende Änderungen:

- Bereitstellen eines neuen geheimen Schlüssels
  - Ein solcher Schlüssel kann mit dem JDK-Dienstprogramm "keytook" generiert werden; siehe 12.4, „Vorgehensweise beim Erstellen eines neuen Keystore”, auf Seite 46
    - Dieser geheime Schlüssel sollte in einem separaten Keystore gespeichert werden.
    - Die Eigenschaften für diese Änderungen am geheimen Schlüssel erfolgen wie in 12.3, „Schlüsselmanagement”, auf Seite 46 beschrieben.
- Bereitstellen neuer Digest-Einstellungen
  - Neue Digest-Einstellungen können einen neuen Salt, einen Iterationszähler bzw. einen Algorithmus enthalten.
    - Die Eigenschaften für diese Digest-Einstellungen erfolgen wie in 4.6, „Cúram-Digest-Eigenschaften”, auf Seite 21 und 12.6, „Vorgehensweise beim Angeben eines Digest Salt”, auf Seite 47 beschrieben (Prozess wie in 12.7, „Vorgehensweise beim Verwenden der ersetzten Digest-Einstellungen für einen Migrationszeitraum”, auf Seite 48 beschrieben).

Denken Sie daran, Ihre Konfigurationsdateien von Personen fernzuhalten, die keinen Zugriff erhalten dürfen. Bewahren Sie insbesondere Entwicklungs-, Test- und Produktionskonfigurationsdaten isoliert auf.

Kapitel 13. Anpassen von Anwendungen für externe Benutzer

13.1 Übersicht

Da externe Benutzer anders als interne Benutzer verarbeitet werden, ist speziell für externe Benutzer eine separate IBM Cúram Social Program Management-Webanwendung erforderlich.

13.2 Erstellen einer Anwendung für externe Benutzer

Für externe Benutzer muss eine neue Web-Client-Anwendung entwickelt werden. Im Referenzhandbuch zu *Cúram Web Client* finden Sie Details zur Erstellung einer neuen Web-Client-Anwendung.

13.3 Erstellen einer Anmeldeseite für externe Benutzerclients

Für eine Anwendung für externe Benutzer muss eine neue Datei des Typs "logon.jsp" erstellt werden. Social Program Management Platform wird mit einer Standardanmeldeseite (`logon.jsp`) ausgeliefert; diese befindet sich im Verzeichnis `lib/curam/web/jsp` von CDEJ (Client Development Environment for Java). Diese Datei muss in einen Ordner des Typs `webclient/components/<custom component>/WebContent` in der Web-Client-Anwendung kopiert und wie folgt modifiziert werden:

Das Element `table` sollte so erweitert werden, dass ein Feld für die verdeckte Eingabe vom Typ user_type einbezogen wird.

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

Dabei weist `EXTERNAL` auf den Typ des externen Benutzers. Dieser Typ kann auf einen beliebigen Wert gesetzt werden, mit Ausnahme von `INTERNAL`.

13.4 Erstellen einer automatischen Anmeldeseite für externe Benutzerclients

Für einige externe Benutzerclientanwendungen ist keine Benutzerauthentifizierung erforderlich; folglich sollten kein Benutzername und kein Kennwort angefordert werden. Es ist nicht möglich, die Authentifizierung in IBM Cúram Social Program Management zu inaktivieren; für diese Voraussetzung ist es daher das Einfachste, ein automatisches Anmeldescript zu schreiben.

Das automatische Anmeldescript verwendet einen fest codierten Benutzernamen und ein entsprechendes Kennwort und stellt diese Angaben als Authentifizierungsinformationen bereit, wenn es eine entsprechende Anforderung gibt. Dies bedeutet, dass alle Benutzer einer solchen Anwendung immer unter demselben Benutzernamen agieren. Die Verwendung eines solchen Scripts muss auf wahre Open Access-Anwendungen beschränkt sein.

Bei der Implementierung von Anwendungen, für die eine automatische Anmeldung erforderlich ist, müssen die Auswirkungen auf das Sitzungsmanagement berücksichtigt werden. Beim Sitzungsmanagement von IBM Cúram Social Program Management werden die Sitzungsinformationen eines Benutzers gepflegt, um bei der Wiederanmeldung des Benutzers die relevanten Sitzungsinformationen zu gewährleisten; d. h., die Registerkarten und die Navigationsstrukturen werden an der Stelle geöffnet, an der der Benutzer sie verlassen hat. Im Fall eines Benutzers, der automatisch angemeldet wurde, müssen diese Informationen nicht gepflegt werden; daher muss das Sitzungsmanagement in diesem Szenario möglicherweise ausgeschaltet werden. Im Referenzhandbuch zu *Cúram Web Client* finden Sie weitere Details zum Ausschalten des Sitzungsmanagements.

Im Folgenden sehen Sie Beispiele für JSP-Scripts zur automatischen An- und Abmeldung.

Anmerkung: Sicherheitsimplementierungen und -konfigurationen sind von Anwendungsserveranbieter zu Anwendungsserveranbieter unterschiedlich; daher funktionieren diese Beispiele möglicherweise nicht in allen Fällen bzw. für alle Anwendungsserverversionen.

```
<?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>
```

Abbildung 8. JSP für automatische Anmeldung

```
<?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>
```

Abbildung 9. JSP für automatische Abmeldung

13.5 Erweitern der Klasse für Benutzer mit öffentlicher Zugriffsberechtigung

Um die angepasste Lösung in der Anwendung zu „verankern", muss die abstrakte Klasse `curam.util.security.PublicAccessUser` erweitert werden. Dazu ist die Implementierung der Schnittstelle `curam.util.security.ExternalAccessSecurity` erforderlich. Diese konkrete Klasse wird während des Authentifizierungs- und Berechtigungsprozesses durchgeführt, um erforderliche Informationen zu dem externen Benutzer zu ermitteln. Diese Klasse und ihre zugehörigen Methoden werden unten ausführlich beschrieben.

13.6 Authentifizieren eines externen Benutzers

Mit der Methode `authenticate()` wird ein externer Benutzer authentifiziert. Diese Methode wird während des Authentifizierungsprozesses aufgerufen, wenn der Benutzer als externer Benutzer identifiziert wird. Im Fall von externen Benutzern wird diese Methode anstelle der konfigurierten Authentifizierung aufgerufen.

Anmerkung: Wenn ein alternativer Authentifizierungsmechanismus konfiguriert ist, z. B. LDAP, müssen sich die externen Benutzer für diesen Mechanismus authentifizieren können.

```
/**
   * Durch die Implementierung dieser Methode sollen die ID und
   * das Kennwort geprüft sowie das Ergebnis der Überprüfung zurückgegeben werden. Wenn die Informationen
   * gültig sind, soll der Code "SecurityStatus.LOGIN" der Codetabelle zurückgegeben werden.
   *
   * @param identifier Die ID des externen Benutzers.
   * @param password Das Kennwort als Zeichenfolge.
   * @param userType Der Typ des externen Benutzers.
   *
   * @return Der Status der Authentifizierung in Form eines Codetabellencodes.
   *
   * @throws AppException - Signatur für generische Ausnahme.
   * @throws InformationalException - Signatur für generische Ausnahme.
   */

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

Die Eingabeparameter für die Methode umfassen eine ID, das mittels Digest verschlüsselte Kennwort als Zeichenfolge und den Typ des zu authentifizierenden externen Benutzers.

Der Parameter `userType` soll mehrere Typen externer Benutzer unterstützen, für die unterschiedliche Authentifizierungsmechanismen erforderlich sind. Die Verwendung dieses Parameters ist von der angepassten Implementierung abhängig.

Das erwartete Ergebnis dieser Methode soll ein Eintrag aus der Codetabelle `curam.util.codetable.SECURITYSTATUS` sein. Im Fall einer erfolgreichen Authentifizierung muss das Ergebnis wie folgt lauten:

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

Für Authentifizierungsfehler enthält diese Codetabelle eine Reihe von Einträgen, darunter `BADUSER`, `BADPWD` und `PWDEXPIRED`. Diese Codetabelle kann so erweitert werden, dass angepasste Codes, wie im Handbuch *Cúram Server Developer* beschrieben, einbezogen werden.

Das durch diese Methode zurückgegebene Authentifizierungsergebnis wird automatisch in der Datenbanktabelle `AuthenticationLog` protokolliert. Weitere Informationen zu dieser Tabelle finden Sie im Handbuch *Cúram Server Developer*.

Mit der abstrakten Klasse `PublicAccessUser` werden außerdem die folgenden abstrakten Methoden definiert, die jede konkrete Unterklasse implementieren muss:

- Die Methode `upgradeSafePasswordValidation()` ist erforderlich, um Kennwörter zu vergleichen, und wird wie folgt definiert:

```
public final boolean upgradeSafePasswordValidation(
final String userName,
final String storedPasswordHash,
final String plaintextPassword)
```

- Mit der Methode `setPassword()` kann das Implementierungselement bei einer Aktualisierung der Entschlüsselung das Kennwort (z. B. ein neues Kennwort) als persistent definieren. Daher wird diese Methode abgerufen, wenn die Methode `upgradeSafePasswordValidation()` abgerufen wird. Die Methodendefinition lautet wie folgt:

```
public abstract void setPassword(String username, String hashedPassword)
throws AppException, InformationalException;
```

Weitere Informationen zu den oben beschriebenen Methoden finden Sie in der zugehörigen JavaDoc der Klasse `PublicAccessUser`.

13.7 Ermitteln von Details zu externen Benutzern

Details für einen externen Benutzer werden durch Aufrufen der Methode `getLoginDetails()` der Schnittstelle `curam.util.security.ExternalAccessSecurity` abgerufen. Diese Details werden direkt nach der Authentifizierung zurückgegeben, um den externen Benutzer an die richtige Startseite zu leiten.

```
/**
 * Durch die Implementierung dieser Methode sollten die Details
 * des Benutzers abgerufen werden, die für dessen Weiterleitung zur richtigen
 * Anwendungsseite erforderlich sind. Zu diesen Informationen gehören der Name
 * der Anwendungsstartseite für den Benutzer, die Standardländereinstellung
 * für den Benutzer sowie eine Liste der Warnungen/Nachrichten, die sich an den Benutzer richten.
 *
 * @param identifier Die ID des externen Benutzers.
 *
 * @return Die Benutzerdetails, einschließlich der
 *         Anwendungsstartseite.
 *
 * @throws AppException - Signatur für generische Ausnahme.
 * @throws InformationalException - Signatur für generische Ausnahme.
 */
UserLoginDetails getLoginDetails(String identifier)
  throws AppException, InformationalException;
```

Durch diese Methode muss eine Instanz der Klasse `curam.util.security.UserLoginDetails` erstellt und zurückgegeben werden. Durch die Verwendung dieser Klasse sollten die folgenden Informationen zurückgegeben werden:

- `UserLoginDetails . setApplicationCode(String code)`

  Der Code, der der Anwendungsstartseite für externe Benutzer entspricht.

  Dabei muss es sich um einen gültigen Eintrag in der Codetabelle `APPLICATION_CODE` handeln.
- `UserLoginDetails . setDefaultLocale(String defaultLocale)`

  Die Standardländereinstellung für den externen Benutzer.

  Dabei handelt es sich um die Ländereinstellung, in der die Anwendung dem externen Benutzer standardmäßig angezeigt wird.
- `UserLoginDetails . addInformationals(InformationalManager informationalManager)`

  Sämtliche Informationen, die dem externen Benutzer angezeigt werden müssen.

  Mit der Klasse `curam.util.exception.InformationalManager` kann eine Reihe von Informations- oder Warnnachrichten erstellt werden, die bei der Anmeldung des externen Benutzers angezeigt werden. Beispiel: Eine Warnung, in der der externe Benutzer darüber informiert wird, dass sein Kennwort demnächst abläuft.

13.8 Berechtigen eines externen Benutzers

Die Methode `getSecurityRole()` wird bei der Berechtigung verwendet, um die dem externen Benutzer zugeordnete Sicherheitsrolle festzulegen. Die für externe Benutzer verwendeten Sicherheitsrollen werden wie die Sicherheitsrollen für interne Benutzer konfiguriert.

```
/**
 * Durch die Implementierung dieser Methode soll die dem
 * externen Benutzer für Berechtigungszwecke zugeordnete Sicherheitsrolle
 * zurückgegeben werden. Wenn der Benutzer nicht existiert, soll "null"
 * zurückgegeben werden.
 *
 * @param identifier Die ID des externen Benutzers.
 *
 * @return Die Sicherheitsrolle für die Berechtigung.
 *
 * @throws AppException - Signatur für generische Ausnahme.
 * @throws InformationalException - Signatur für generische Ausnahme.
 */
String getSecurityRole(String identifier)
  throws AppException, InformationalException;
```

SDEJ ruft während des Berechtigungsprozesses eine Implementierung dieser Methode auf, wenn der Benutzer nicht im Sicherheitscache existiert. Im Sicherheitscache dürfen nur interne Benutzer vorhanden sein. Dies bedeutet, dass die für die Identifizierung externer Benutzer verwendeten IDs eindeutig sein müssen und nicht mit Benutzernamen kollidieren dürfen, die für interne Benutzer eingerichtet wurden, es sei denn, die in 7.3, „Benutzerbereich”, auf Seite 29 beschriebene angepasste Schnittstelle `UserScope` wurde implementiert. Ansonsten werden, falls es zu Benutzernamenskonflikten kommt, die dem internen Benutzer zugewiesenen Zugriffsrechte auch für den externen Benutzer verwendet.

Falls für den externen Benutzer keine Rolle bestimmt werden kann, muss "null" zurückgegeben werden, damit SDEJ den Berechtigungsfehler ordnungsgemäß melden kann.

13.9 Ermitteln des Benutzertyps

Mit der Methode `getUserType()` wird festgestellt, ob es sich bei einem Benutzer um einen externen Benutzer handelt.

```
/**
 * Es wird der Typ des Benutzers zurückgegeben. Dadurch wird die Unterstützung für
 * unterschiedliche Typen von externen Benutzern ermöglicht. Wenn es nur einen einzigen
 * Typ von externem Benutzer gibt, wird nur "EXTERNAL" zurückgegeben.
 *
 * @param identifier Die ID des externen Benutzers.
 *
 * @return Der Typ des externen Benutzers.
 *
 * @throws AppException - Signatur für generische Ausnahme.
 * @throws InformationalException - Signatur für generische Ausnahme.
 */
 String getUserType(final String identifier)
   throws AppException, InformationalException;
```

Diese Methode wird mit der Methode `getProgramUserType()` in `curam.util.transaction.TransactionInfo` aufgerufen, um den Typ von Benutzer zurückzugeben, sollte der Benutzer nicht als interner Benutzer erkannt worden sein. Für interne Benutzer wird immer „INTERNAL” zurückgegeben.

Für externe Benutzer kann es mehrere Typen von externen Benutzern geben; durch diese Methode sollte also der jeweilige Typ von externem Benutzer zurückgegeben werden.

13.10 Verhindern der Löschung einer Sicherheitsrolle: Rollennutzungszahl

Mit der Methode `getRoleUsageCount()` wird die Löschung einer Sicherheitsrolle verhindert, die momentan von einem externen Benutzer referenziert wird.

```
/**
 * Gibt die Anzahl der Benutzer zurück, die eine bestimmte Rolle verwenden. Diese
 * Methode wird verwendet, um sicherzustellen, dass eine Rolle nicht gelöscht werden kann,
 * wenn sie von einem externen Benutzer verwendet wird.
 *
 * @param role Der Name der Sicherheitsrolle.
 *
 * @return Die Anzahl der Benutzer, die die angegebene Rolle derzeit
 *         verwenden.
 *
 * @throws AppException - Signatur für generische Ausnahme.
 * @throws InformationalException - Signatur für generische Ausnahme.
 */
 int getRoleUsageCount(String role)
   throws AppException, InformationalException;
```

Sicherheitsrollen, die von einem beliebigen Benutzer referenziert werden (interner oder externer Benutzer), können nicht entfernt werden. Mit dieser Methode sollte die Zahl 1 oder höher zurückgegeben werden, wenn eine beliebige Anzahl von Benutzern die angegebene Rolle referenziert.

13.11 Abrufen eines registrierten Benutzernamens

Mit der Methode `getRegisteredUserName()` wird der Benutzername in korrekter Groß-/Kleinschreibung abgerufen; dieser kann sich von dem bei der Anmeldung eingegebenen Benutzernamen unterscheiden.

```
/**
 * Es wird die korrekte Groß-/Kleinschreibung für diesen Benutzer abgerufen, unabhängig davon,
 * ob von dem angemeldeten Benutzer Buchstaben in gemischter Schreibweise eingegeben wurden.
 *
 * @param identifier Die ID des externen Benutzers,
 * deren Groß-/Kleinschreibung möglicherweise nicht mit der dauerhaft festgelegten ID für den Benutzer
 * übereinstimmt.
 *
 * @return Die eigentliche Groß-/Kleinschreibung für diesen Benutzer, bevor diese
 * durch externe Faktoren geändert wurde.
 *
 * @throws AppException - Signatur für generische Ausnahme.
 * @throws InformationalException - Signatur für generische Ausnahme.
 */
 public String getRegisteredUserName(final String identifier)
   throws AppException, InformationalException;
```

Mit der Standardimplementierung für diese Methode muss der angegebene Benutzername zurückgegeben werden. Nur wenn "curam.security.casesensitive" auf "false" gesetzt wurde, kann für diese Methode eine Änderung der Groß-/Kleinschreibung des Benutzernamens erforderlich sein.

Anmerkung: Wenn die Eigenschaft "curam.security.casesensitive" auf "false" gesetzt wurde und für externe Benutzer erforderlich ist, müssen alle Methoden in dieser Schnittstelle die jeweiligen groß-/kleinschreibungsspezifischen Anforderungen berücksichtigen.

13.12 Lesen der Benutzereinstellungen

Mit der Methode `getUserPreferenceSetID()` wird die Gruppen-ID für die Benutzereinstellungen abgerufen, die einem externen Benutzer zugeordnet ist. Wenn es für einen externen Benutzer keine Benutzereinstellungen gibt, werden die Standardvorgaben für ihn verwendet. Weitere Details zu den Benutzereinstellungen finden Sie im Kapitel zu den Benutzereinstellungen im Handbuch *Cúram Server Developer*.

```
/**
 * Mit dieser Methode wird eine Reihe von Benutzereinstellungen abgerufen, die
 * einem externen Benutzer zugeordnet sind. "userPrefSetID" ist ein
 * Fremdschlüssel für die Tabelle "UserPreferenceInfo".
 * In der Tabelle "UserPreferenceInfo" befinden sich Informationen zu den
 * Benutzereinstellungen.
 *
 * @param identifier Die ID des externen Benutzers.
 *
 * @return Die userPrefSetID für den externen Benutzer.
 *
 * @throws AppException - Signatur für generische Ausnahme.
 * @throws InformationalException - Signatur für generische Ausnahme.
 */
 String getUserPreferenceSetID(final String identifier)
   throws AppException, InformationalException;
```

Mit der Standardimplementierung für diese Methode muss die einem externen Benutzer zugeordnete Gruppen-ID für die Benutzereinstellungen zurückgegeben werden.

13.13 Modifizieren der Benutzereinstellungen

Mit der Methode `modifyUserPreferenceSetID()` werden die Details des externen Benutzers mit einer neuen Gruppe von Benutzereinstellungen aktualisiert. In "Benutzereinstellungen" finden Sie weitere Details zu den Benutzereinstellungen.

```
/**
 * Mit dieser Methode werden die Details zum externen Benutzer durch neue Benutzereinstellungen
 * aktualisiert.
 *
 * @param userPreferenceSetID Die ID für die Benutzereinstellungen.
 * @param username Die ID des externen Benutzers.
 *
 * @throws AppException - Signatur für generische Ausnahme.
 * @throws InformationalException - Signatur für generische Ausnahme.
 */
 void modifyUserPreferenceSetID(
  final String userPreferenceSetID, final String username)
     throws AppException, InformationalException;
```

Mit der Standardimplementierung für diese Methode muss die einem externen Benutzer zugeordnete Gruppen-ID für die Benutzereinstellungen aktualisiert werden.

13.14 Konfigurieren der Sicherheit für externen Zugriff

Die Eigenschaft `curam.custom.externalaccess.implementation` muss in der Datei `Application.prx` gesetzt werden, um den vollständig qualifizierten Namen der Klasse anzugeben, die die oben genannten Schnittstelle implementiert.

Anmerkung: Die Eigenschaft "curam.custom.externalaccess.implementation" ist keine dynamische Eigenschaft; wenn sie geändert wird, muss die Anwendung erneut gestartet werden, damit die Änderung wirksam wird.

13.15 Mithilfe der Schnittstelle "UserScope" feststellen, ob ein Benutzer ein interner oder externer Benutzer ist

Zur Unterstützung alternativer Methoden, um festzustellen, ob es sich bei dem Benutzer um einen internen oder externen Benutzer handelt, kann die angepasste Schnittstelle `UserScope` verwendet werden. So kann diese angepasste Schnittstelle z. B. implementiert werden, um auf der Basis zusätzlicher Informationen den Benutzertyp zu ermitteln und um die Notwendigkeit von eindeutigen Namen für externe und interne Benutzer zu entfernen.

Für die Bereitstellung einer angepassten Implementierung für die Ermittlung des Benutzertyps muss die Schnittstelle `curam.util.security.UserScope` implementiert werden. Diese Schnittstelle weist eine einzige Methode auf (`isUserExternal()`), mit der der Benutzertyp ermittelt wird. Diese Methode muss "true" zurückgeben, wenn der Benutzer als extern erachtet wird, bzw. "false", wenn es sich um einen internen Benutzer handelt.

Um anzugeben, dass die angepasste Implementierung verwendet werden soll, muss die Eigenschaft "curam.custom.userscope.implementation" in der Datei `Application.prx` definiert werden. Diese sollte auf den vollständig qualifizierten Namen der Klasse gesetzt sein, die die Schnittstelle `UserScope` implementiert.

Anmerkung: Die Eigenschaft "curam.custom.userscope.implementation" ist keine dynamische Eigenschaft; wenn sie geändert wird, muss die Anwendung erneut gestartet werden, damit die Änderung wirksam wird.

Die Methode "isUserExternal()" der Schnittstelle "UserScope" wird im Folgenden ausführlich beschrieben:

13.16 Ermittlung des Benutzertyps

Die Methode `isUserExternal()` wird in der Anwendung immer dann aufgerufen, wenn der Typ des Benutzers ermittelt werden soll. Dazu gehört, festzustellen, wann sich der Benutzer bei der Anwendung anmeldet und wann er versucht, auf gesicherte Elemente von IBM Cúram Social Program Management zuzugreifen.

```
/**
 * Durch die Implementierung dieser Methode soll der Typ des Benutzers
 * ermittelt werden, der bei der Anwendung angemeldet ist. Es gibt 2 Typen von
 * Benutzern: INTERNE und EXTERNE. Wenn es sich um einen EXTERNEN Benutzer handelt,
 * sollte diese Methode "true" zurückgeben. Wenn "false" zurückgegeben wird,
 * wird der Benutzer als INTERNER Benutzer betrachtet.
 *
 * @param username - Der Benutzername.
 * @return Der boolesche Wert "true" weist auf einen externen Benutzer hin;
 * "false" weist auf einen internen Benutzer hin.
 *
 * @throws AppException - Signatur für generische Ausnahme.
 * @throws InformationalException - Signatur für generische Ausnahme.
 */
boolean isUserExternal(String username)
  throws AppException, InformationalException;
```

Bemerkungen

Die vorliegenden Informationen wurden für Produkte und Services entwickelt, die auf dem deutschen Markt angeboten werden. Möglicherweise bietet IBM die in dieser Dokumentation beschriebenen Produkte, Services oder Funktionen in anderen Ländern nicht an. Informationen über die gegenwärtig im jeweiligen Land verfügbaren Produkte und Services sind beim zuständigen IBM Ansprechpartner erhältlich. Hinweise auf IBM Lizenzprogramme oder andere IBM Produkte bedeuten nicht, dass nur Programme, Produkte oder Services von IBM verwendet werden können. Anstelle der IBM Produkte, Programme oder Services können auch andere, ihnen äquivalente Produkte, Programme oder Services verwendet werden, solange diese keine gewerblichen oder anderen Schutzrechte von IBM verletzen. Die Verantwortung für den Betrieb von Produkten, Programmen und Services anderer Anbieter liegt beim Kunden. Für die in diesem Handbuch beschriebenen Erzeugnisse und Verfahren kann es IBM Patente oder Patentanmeldungen geben. Mit der Auslieferung dieses Handbuchs ist keine Lizenzierung dieser Patente verbunden. Lizenzanforderungen sind schriftlich an folgende Adresse zu richten (Anfragen an diese Adresse müssen auf Englisch formuliert werden):

IBM Director of Licensing
IBM Europe, Middle East & Africa
Tour Descartes
2, avenue Gambetta
92066 Paris La Defense Cedex
France

Trotz sorgfältiger Bearbeitung können technische Ungenauigkeiten oder Druckfehler in dieser Veröffentlichung nicht ausgeschlossen werden. Die hier enthaltenen Informationen werden in regelmäßigen Zeitabständen aktualisiert und als Neuausgabe veröffentlicht. IBM kann ohne weitere Mitteilung jederzeit Verbesserungen und/oder Änderungen an den in dieser Veröffentlichung beschriebenen Produkten und/oder Programmen vornehmen.

Verweise in diesen Informationen auf Websites anderer Anbieter werden lediglich als Service für den Kunden bereitgestellt und stellen keinerlei Billigung des Inhalts dieser Websites dar. Das über diese Websites verfügbare Material ist nicht Bestandteil des Materials für dieses IBM Produkt. Die Verwendung dieser Websites geschieht auf eigene Verantwortung.

Werden an IBM Informationen eingesandt, können diese beliebig verwendet werden, ohne dass eine Verpflichtung gegenüber dem Einsender entsteht. Lizenznehmer des Programms, die Informationen zu diesem Produkt wünschen mit der Zielsetzung: (i) den Austausch von Informationen zwischen unabhängig voneinander erstellten Programmen und anderen Programmen (einschließlich des vorliegenden Programms) sowie (ii) die gemeinsame Nutzung der ausgetauschten Informationen zu ermöglichen, wenden sich an folgende Adresse:

IBM Corporation
Dept F6, Bldg 1
294 Route 100
Somers NY 10589-3216
U.S.A.

Die Bereitstellung dieser Informationen kann unter Umständen von bestimmten Bedingungen - in einigen Fällen auch von der Zahlung einer Gebühr - abhängig sein.

Die Lieferung des in diesem Dokument beschriebenen Lizenzprogramms sowie des zugehörigen Lizenzmaterials erfolgt auf der Basis der IBM Rahmenvereinbarung bzw. der Allgemeinen Geschäftsbedingungen von IBM, der IBM Internationalen Nutzungsbedingungen für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer kontrollierten Umgebung. Die Ergebnisse, die in anderen Betriebsumgebungen erzielt werden, können daher erheblich von den hier erzielten Ergebnissen abweichen. Einige Daten stammen möglicherweise von Systemen, deren Entwicklung noch nicht abgeschlossen ist. Eine Gewährleistung, dass diese Daten auch in allgemein verfügbaren Systemen erzielt werden, kann nicht gegeben werden. Darüber hinaus wurden einige Daten unter Umständen durch Extrapolation berechnet. Die tatsächlichen Ergebnisse können davon abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Alle Informationen zu Produkten anderer Anbieter stammen von den Anbietern der aufgeführten Produkte, deren veröffentlichten Ankündigungen oder anderen allgemein verfügbaren Quellen.

IBM hat diese Produkte nicht getestet und kann daher keine Aussagen zu Leistung, Kompatibilität oder anderen Merkmalen machen. Fragen zu den Leistungsmerkmalen von Produkten anderer Anbieter sind an den jeweiligen Anbieter zu richten.

Aussagen über Pläne und Absichten von IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele von IBM.

Alle von IBM angegebenen Preise sind empfohlene Richtpreise und können jederzeit ohne weitere Mitteilung geändert werden. Händlerpreise können u. U. von den hier genannten Preisen abweichen.

Diese Veröffentlichung dient nur zu Planungszwecken. Die in dieser Veröffentlichung enthaltenen Informationen können geändert werden, bevor die beschriebenen Produkte verfügbar sind.

Diese Veröffentlichung enthält Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufs. Sie sollen nur die Funktionen des Lizenzprogramms illustrieren und können Namen von Personen, Firmen, Marken oder Produkten enthalten. Alle diese Namen sind frei erfunden; Ähnlichkeiten mit tatsächlichen Namen und Adressen sind rein zufällig.

COPYRIGHTLIZENZ:

Diese Veröffentlichung enthält Beispielanwendungsprogramme, die in Quellensprache geschrieben sind und Programmiertechniken in verschiedenen Betriebsumgebungen veranschaulichen. Sie dürfen diese Beispielprogramme kostenlos kopieren, ändern und verteilen, wenn dies zu dem Zweck geschieht, Anwendungsprogramme zu entwickeln, zu verwenden, zu vermarkten oder zu verteilen, die mit der Anwendungsprogrammierschnittstelle für die Betriebsumgebung konform sind, für die diese Beispielprogramme geschrieben werden. Diese Beispiele wurden nicht unter allen denkbaren Bedingungen getestet. Daher kann IBM die Zuverlässigkeit, Wartungsfreundlichkeit oder Funktion dieser Programme weder zusagen noch gewährleisten. Die Beispielprogramme werden ohne Wartung (auf "as-is"-Basis) und ohne jegliche Gewährleistung zur Verfügung gestellt. IBM übernimmt keine Haftung für Schäden, die durch die Verwendung der Beispielprogramme entstehen.

Kopien oder Teile der Beispielprogramme bzw. daraus abgeleiteter Code müssen folgenden Copyrightvermerk beinhalten:

© (Name Ihrer Firma) (Jahr). Teile des vorliegenden Codes wurden aus Beispielprogrammen der IBM Corporation abgeleitet.

© Copyright IBM Corp. _Jahreszahl oder Jahreszahlen eingeben_. Alle Rechte vorbehalten.

Informationen zu Programmierschnittstellen

In der vorliegenden Veröffentlichung werden vorgesehene Programmierschnittstellen dokumentiert, mit deren Hilfe Kunden Programme für den Zugriff auf IBM Cúram Social Program Management-Services schreiben können.

Marken

IBM, das IBM Logo und ibm.com sind Marken oder eingetragene Marken der International Business Machines Corp. in den USA und/oder anderen Ländern. Weitere Produkt- und Servicenamen können Marken von IBM oder anderen Unternehmen sein. Eine aktuelle Liste der IBM Marken finden Sie auf der Webseite "Copyright and trademark information" unter http://www.ibm.com/legal/us/en/copytrade.shtml.

Apache ist eine Marke von Apache Software Foundation.

Oracle, WebLogic Server, Java und alle auf Java basierenden Marken und Logos sind eingetragene Marken der Oracle Corporation und/oder ihrer verbundenen Unternehmen.

Andere Namen können Marken der jeweiligen Rechtsinhaber sein. Weitere Firmen-, Produkt- und Servicenamen können Marken oder Servicemarken anderer Unternehmen sein.