

IBM Cúram Social Program Management



Guide de sécurité Cúram

Version 6.0.5

IBM Cúram Social Program Management



Guide de sécurité Cúram

Version 6.0.5

Important

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales à la section «Remarques», à la page 61

LE PRESENT DOCUMENT EST LIVRE EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE.

Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. Les informations qui y sont fournies sont susceptibles d'être modifiées avant que les produits décrits ne deviennent eux-mêmes disponibles. En outre, il peut contenir des informations ou des références concernant certains produits, logiciels ou services non annoncés dans ce pays. Cela ne signifie cependant pas qu'ils y seront annoncés.

Pour plus de détails, pour toute demande d'ordre technique, ou pour obtenir des exemplaires de documents IBM, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial.

Vous pouvez également consulter les serveurs Internet suivants :

- <http://www.fr.ibm.com> (serveur IBM en France)
- <http://www.can.ibm.com> (serveur IBM au Canada)
- <http://www.ibm.com> (serveur IBM aux Etats-Unis)

*Compagnie IBM France
Direction Qualité
17, avenue de l'Europe
92275 Bois-Colombes Cedex*

Cette édition s'applique à IBM Cúram Social Program Management v6.0 5 et à toutes les révisions suivantes, sauf indication contraire dans de nouvelles éditions.

Eléments sous licence - Propriété d'IBM.

© Copyright IBM Corporation 2012, 2013.

© Cúram Software Limited. 2011. All rights reserved.

Table des matières

Figures	v	4.7 Mots de passe chiffrés	22
Tableaux	vii	Chapitre 5. Mise en cache des données de sécurité	23
Avis aux lecteurs canadiens.	ix	5.1 Présentation	23
Chapitre 1. Sécurité Cúram	1	5.2 Cache de sécurité Cúram.	23
1.1 Objet.	1	5.3 Actualisation du cache	23
1.2 Public concerné	1	5.4 Echec d'actualisation du cache	23
1.3 Présentation	1	5.5 Comportement de la mise en cache WebSphere	23
1.4 Chapitres	2	Chapitre 6. Sécurité des autres clients 25	
Chapitre 2. Authentification	3	6.1 Présentation	25
2.1 Présentation	3	6.2 Utilisateurs Cúram obligatoires	25
2.2 Authentification	3	6.3 Services Web.	25
2.3 Architecture d'authentification	4	6.4 Traitement par lots.	26
2.4 Authentification par défaut	4	6.5 Messagerie JMS.	26
2.5 Autres ID de connexion	5	6.6 Traitement différé	27
2.6 Page de connexion	6	Chapitre 7. Applications d'utilisateurs externes	29
2.7 Personnalisation de la page de connexion	7	7.1 Présentation	29
2.8 Module de connexion JAAS Cúram	7	7.2 Applications d'utilisateurs externes	29
2.9 Gestion des mots de passe	8	7.3 Portée d'utilisateur.	29
2.10 Configuration par défaut de WebLogic Server.	8	7.4 Déploiement d'une application externe	30
2.11 Configuration par défaut de WebSphere.	8	Chapitre 8. Utilisation d'une connexion unique	33
2.12 Personnalisation du module de connexion JAAS	11	8.1 Présentation	33
2.13 Processus de vérification pour l'authentification	11	8.2 Connexion unique avec WebSphere	33
2.14 Authentification par défaut	11	8.3 Connexion unique avec WebLogic Server	34
2.15 Processus de vérification par défaut.	11	Chapitre 9. Autres remarques relatives à la sécurité	35
2.16 Tentatives d'authentification	12	9.1 Présentation	35
2.17 Personnalisation de l'authentification par défaut	12	9.2 Paramètres SSL pour l'application.	35
2.18 Authentification d'identité uniquement.	12	Chapitre 10. Personnalisation de l'authentification	37
2.19 Personnalisation de l'authentification d'identité uniquement	13	10.1 Personnalisation de la page de connexion.	37
2.20 Authentification de sécurité d'accès externe	14	10.2 Application d'un style à la page de connexion	37
2.21 Vérifications personnalisées	14	10.3 Activation des noms d'utilisateurs avec caractères étendus pour WebLogic Server	37
Chapitre 3. Autorisation	15	10.4 Changement de la sensibilité à la casse du nom d'utilisateur	37
3.1 Présentation	15	10.5 Ajout de vérifications personnalisées au processus d'authentification	37
3.2 Utilisateurs, rôles et groupes	15	10.6 Configuration de l'authentificateur personnalisé	38
3.3 Identificateurs de sécurité	16	10.7 Configuration de l'authentification d'identité uniquement	38
3.4 Identificateurs de fonction	16	10.8 Ajout de l'interface de rappel d'échec de l'actualisation du cache	38
3.5 Identificateurs de sécurité de niveau zone	16	10.9 Désactivation des paramètres SSL de l'application	38
3.6 Identificateurs de sécurité définis par l'utilisateur	16		
3.7 Autorisation d'exécution	17		
3.8 Vérifications d'autorisation client	17		
3.9 Vérifications d'autorisation de serveur	17		
Chapitre 4. Cryptographie dans Cúram 19			
4.1 Présentation	19		
4.2 Chiffrement	19		
4.3 Traitement	19		
4.4 Propriétés de cryptographie.	20		
4.5 Paramètres de chiffrement Cúram.	20		
4.6 Paramètres de traitement Cúram	21		

10.10 Modification du fichier web.xml pour l'application client	39
10.11 Modification de la configuration du serveur d'applications.	39
10.12 Analyse de la table de données AuthenticationLog	40

Chapitre 11. Personnalisation d'autorisation 41

11.1 Présentation.	41
11.2 Création de mappage de données d'autorisation	41
11.3 Création d'un nouveau rôle de sécurité	41
11.4 Création d'un nouveau groupe de sécurité	41
11.5 Liaison du groupe de sécurité au rôle de sécurité.	41
11.6 Création de l'identificateur de sécurité	42
11.7 Liaison du groupe de sécurité à l'identificateur de sécurité.	42
11.8 Liaison du rôle de sécurité à l'utilisateur	42
11.9 Chargement des informations de sécurité sur la base de données.	42
11.10 Création d'identificateurs de fonction	42
11.11 Désactivation de la sécurité pour une méthode de processus	43
11.12 Remarques sur la sécurité au cours du développement	43
11.13 Contrôle de la consignation des échecs d'autorisation du client	43
11.14 Autorisation de nouveaux types d'identificateurs de sécurité	44
11.15 Analyse de la table de base de données AuthorisationLog	44

Chapitre 12. Personnalisation de la cryptographie. 45

12.1 Présentation	45
12.2 Personnalisation du chiffrement	45

12.3 Gestion des clés	45
12.4 Création d'un nouveau fichier de clés	46
12.5 Personnalisation du traitement	47
12.6 Spécification d'un sel de traitement	47
12.7 Utilisation des paramètres de traitement obsolètes pour une période de migration	48
12.8 Modification de votre configuration de cryptographie pour un système de production.	49

Chapitre 13. Personnalisation des applications d'utilisateurs externes . . . 51

13.1 Présentation	51
13.2 Création d'une application d'utilisateur externe	51
13.3 Création d'une page de connexion client d'utilisateur externe	51
13.4 Création d'une page de connexion automatique de client de l'utilisateur externe.	51
13.5 Extension de la classe d'utilisateurs d'accès public	53
13.6 Authentification d'un utilisateur externe	53
13.7 Détermination des détails de l'utilisateur externe	55
13.8 Autorisation d'un utilisateur externe	55
13.9 Détermination du type d'utilisateur.	56
13.10 Prévention de la suppression d'un rôle de sécurité : Nombre d'utilisations du rôle	56
13.11 Récupération d'un nom d'utilisateur enregistré	57
13.12 Relevé des préférences utilisateur	57
13.13 Modification des préférences utilisateur	58
13.14 Configuration de la sécurité d'accès externe.	58
13.15 Détermination d'un utilisateur interne ou externe à l'aide de l'interface UserScope	58
13.16 Détermination du type d'utilisateur	59

Remarques 61

Documentation sur l'interface de programmation.	63
Marques	63

Figures

1. Architecture d'authentification	4	5. Authentification d'identité uniquement	13
2. Authentification par défaut.	5	6. Prise en charge des connexions avec caractères étendus par WebLogic Server	37
3. Flux d'authentification par défaut de WebSphere.	10	7. Exemple d'utilisation d'isSIDAuthorised()	44
4. Flux d'authentification de WebSphere avec le registre d'utilisateurs activé	10	8. JSP de connexion automatique	52
		9. JSP de déconnexion automatique	53

Tableaux

- | | | | | | |
|----|--|----|----|--|----|
| 1. | Contenus du journal d'authentification . . . | 40 | 3. | Relation des arguments de commande keytool
dans les propriétés de cryptographie Cúram . | 46 |
| 2. | Contenus du journal d'authentification . . . | 44 | | | |

Avis aux lecteurs canadiens

Le présent document a été traduit en France. Voici les principales différences et particularités dont vous devez tenir compte.

Illustrations

Les illustrations sont fournies à titre d'exemple. Certaines peuvent contenir des données propres à la France.

Terminologie

La terminologie des titres IBM peut différer d'un pays à l'autre. Reportez-vous au tableau ci-dessous, au besoin.

IBM France	IBM Canada
ingénieur commercial	représentant
agence commerciale	succursale
ingénieur technico-commercial	informaticien
inspecteur	technicien du matériel

Claviers

Les lettres sont disposées différemment : le clavier français est de type AZERTY, et le clavier français-canadien de type QWERTY.








OS/2 et Windows - Paramètres canadiens

Au Canada, on utilise :

- les pages de codes 850 (multilingue) et 863 (français-canadien),
- le code pays 002,
- le code clavier CF.

Nomenclature

Les touches présentées dans le tableau d'équivalence suivant sont libellées différemment selon qu'il s'agit du clavier de la France, du clavier du Canada ou du clavier des États-Unis. Reportez-vous à ce tableau pour faire correspondre les touches françaises figurant dans le présent document aux touches de votre clavier.

France	Canada	Etats-Unis
 (Pos1)		Home
Fin	Fin	End
 (PgAr)		PgUp
 (PgAv)		PgDn
Inser	Inser	Ins
Suppr	Suppr	Del
Echap	Echap	Esc
Attn	Intrp	Break
Impr écran	ImpEc	PrtSc
Verr num	Num	Num Lock
Arrêt défil	Défil	Scroll Lock
 (Verr maj)	FixMaj	Caps Lock
AltGr	AltCar	Alt (à droite)

Brevets

Il est possible qu'IBM détienne des brevets ou qu'elle ait déposé des demandes de brevets portant sur certains sujets abordés dans ce document. Le fait qu'IBM vous fournisse le présent document ne signifie pas qu'elle vous accorde un permis d'utilisation de ces brevets. Vous pouvez envoyer, par écrit, vos demandes de renseignements relatives aux permis d'utilisation au directeur général des relations commerciales d'IBM, 3600 Steeles Avenue East, Markham, Ontario, L3R 9Z7.

Assistance téléphonique

Si vous avez besoin d'assistance ou si vous voulez commander du matériel, des logiciels et des publications IBM, contactez IBM direct au 1 800 465-1234.

Chapitre 1. Sécurité Cúram

1.1 Objet

L'objectif de ce guide est de décrire les aspects de sécurité à prendre en compte lors du développement et du déploiement d'une application d'entreprise IBM Cúram Social Program Management. Le terme de sécurité est utilisé pour décrire plusieurs domaines différents. Dans le cadre de ce document, les domaines suivants sont couverts : authentification et autorisation. Ce guide décrit également les éléments sécurisables des applications IBM Cúram Social Program Management.

Ce guide est divisé en deux parties. La première partie contient une présentation des domaines de sécurité et doit être lue par toutes les parties intéressées, à savoir les architectes techniques et les développeurs. La seconde partie fournit des informations pratiques sur le développement ainsi que des exemples de sécurité d'application.

1.2 Public concerné

Il existe deux types principaux de public concerné par ce document :

- Les architectes techniques qui ont besoin d'envisager l'intégration avec d'autres systèmes lors de la phase de déploiement, par exemple avec le protocole LDAP.
- Les développeurs qui doivent déterminer le type d'application à développer, par exemple si l'application est destinée à des utilisateurs internes, externes ou les deux.

Remarque : Les utilisateurs internes sont des utilisateurs qui existent sur la table de base de données des utilisateurs Cúram. Ils font partie de l'organisation et servent généralement à gérer les demandes des participants. Les utilisateurs externes correspondent à tous les autres types d'utilisateurs. Les utilisateurs externes ne font pas partie de l'organisation. Leur accès est limité. Un utilisateur externe peut être par exemple un fournisseur qui propose un service à l'organisation.

1.3 Présentation

La sécurité est intégrée à l'infrastructure qui soutient le développement de l'application IBM Cúram Social Program Management. Elle prend en charge l'authentification d'un utilisateur lors de la connexion et assure également le support du processus d'autorisation. La conception d'une application n'est pas terminée tant que les implications liées à la sécurisation de l'application contre l'accès non autorisé aux données sensibles ou aux fonctionnalités ne sont pas envisagées. La sécurité est donc l'une des priorités essentielles au cours du développement d'application.

Voici les principaux concepts de sécurité d'IBM Cúram Social Program Management :

- Authentification
- Autorisation

Il convient de noter également le concept de sécurité des localisations. Au niveau de l'organisation, la sécurité des localisations limite l'accès de l'utilisateur aux informations des clients et dossiers. La sécurité des données de localisation peut aussi être configurée afin que l'utilisateur puisse accéder à d'autres localisations que les siennes. Ce guide ne fournit pas d'informations détaillées sur la sécurité des localisations. Pour plus d'informations sur la sécurité des localisations, consultez le *Guide d'administration des localisations Cúram*.

1.4 Chapitres

Ce guide comprend deux parties. La partie 1 comprend les chapitres suivants, qui fournissent une présentation générale de l'architecture de sécurité IBM Cúram Social Program Management pour l'application et le déploiement sur des serveurs d'applications :

Chapitre 2. Authentification

Ce chapitre décrit l'architecture d'authentification pour IBM Cúram Social Program Management, en fournissant une description détaillée de chaque domaine et des points personnalisables disponibles.

Chapitre 3. Autorisation

Ce chapitre décrit le fonctionnement de l'autorisation et la manière dont celle-ci peut être configurée pour les utilisateurs.

Chapitre 4 Cryptographie dans Cúram

Ce chapitre décrit comment IBM Cúram Social Program Management utilise la cryptographie pour sécuriser les mots de passe.

Chapitre 5 Mise en cache des données de sécurité

Ce chapitre décrit le cache de sécurité Cúram intégré, ainsi que le cache IBM® WebSphere Application Server et explique en quoi ceux-ci affectent l'authentification d'utilisateur.

Chapitre 6 Sécurité des autres clients

Ce chapitre décrit les noms d'utilisateurs qui doivent exister sur la table de base de données des utilisateurs Cúram pour garantir que les processus tels que le flux de travaux puissent s'exécuter correctement.

Chapitre 7 Applications d'utilisateurs externes

Ce chapitre indique les raisons pour lesquelles une application d'utilisateur externe est nécessaire et les éléments à prendre en compte pour cette dernière.

Chapitre 8 Utilisation d'une connexion unique

Ce chapitre décrit les propriétés du serveur d'applications à prendre en compte lorsqu' IBM Cúram Social Program Management est utilisé dans une solution de connexion unique.

Chapitre 9 Autres remarques relatives à la sécurité

Ce chapitre fournit une courte présentation de certaines pratiques et des remarques relatives à la sécurité externe.

La partie 2 comprend les chapitres suivants, qui proposent plusieurs guides pratiques concernant les activités de développement pour coder et personnaliser la sécurité :

Chapitre 10 Personnalisation de l'authentification

Ce chapitre décrit les points de personnalisation et les artefacts de développement pertinents pour l'authentification.

Chapitre 11 Personnalisation de l'autorisation

Ce chapitre décrit comment implémenter l'autorisation pour IBM Cúram Social Program Management.

Chapitre 12 Personnalisation de la cryptographie

Ce chapitre décrit comment personnaliser l'utilisation de la cryptographie dans Cúram.

Chapitre 13 Personnalisation des applications d'utilisateurs externes

Ce chapitre décrit la méthode de développement d'une application d'utilisateur externe.

Chapitre 2. Authentification

2.1 Présentation

Ce chapitre traite de l'authentification pour IBM Cúram Social Program Management. L'authentification consiste à déterminer si un utilisateur est la personne qu'il prétend être. L'authentification est nécessaire lorsqu'un utilisateur doit être contrôlé afin de pouvoir accéder à une ressource sécurisée sur un système.

Dans l'authentification par formulaire, un utilisateur dispose d'un formulaire qui lui permet d'entrer des données d'identification par nom d'utilisateur et par mot de passe. Ces données d'identification sont comparées avec les données d'identification stockées sur le système pour ce nom d'utilisateur. Si elles correspondent, l'utilisateur est considéré comme utilisateur authentifié pour le système. Pour des raisons de sécurité, il est recommandé que le mot de passe servant à authentifier un utilisateur soit stocké sur le système sous une forme chiffrée.

Le client Web IBM Cúram Social Program Management est configuré pour prendre en charge l'authentification par formulaire, ce qui signifie qu'avant de pouvoir accéder à un contenu de client Web, un utilisateur est redirigé vers un formulaire de connexion pour l'authentification.

Le processus d'authentification implique la vérification du nom d'utilisateur et le mot de passe, et cette action est effectuée par défaut par un module de connexion JAAS (Java™ Authentication and Authorization Service). HTTPS/SSL est activé par défaut sur le client Web, ce qui garantit que le mode d'authentification de connexion par formulaire est sécurisé.

2.2 Authentification

Différents modes d'authentification peuvent être configurés (en fonction des exigences d'authentification) via le module de connexion JAAS Cúram.

Voici les modes d'authentification pris en charge :

- Authentification par défaut ;
- Authentification d'identité uniquement ;
- Authentification de sécurité d'accès externe.

Chacun de ces modes est décrit en détail dans les sections ci-dessous.

2.3 Architecture d'authentification

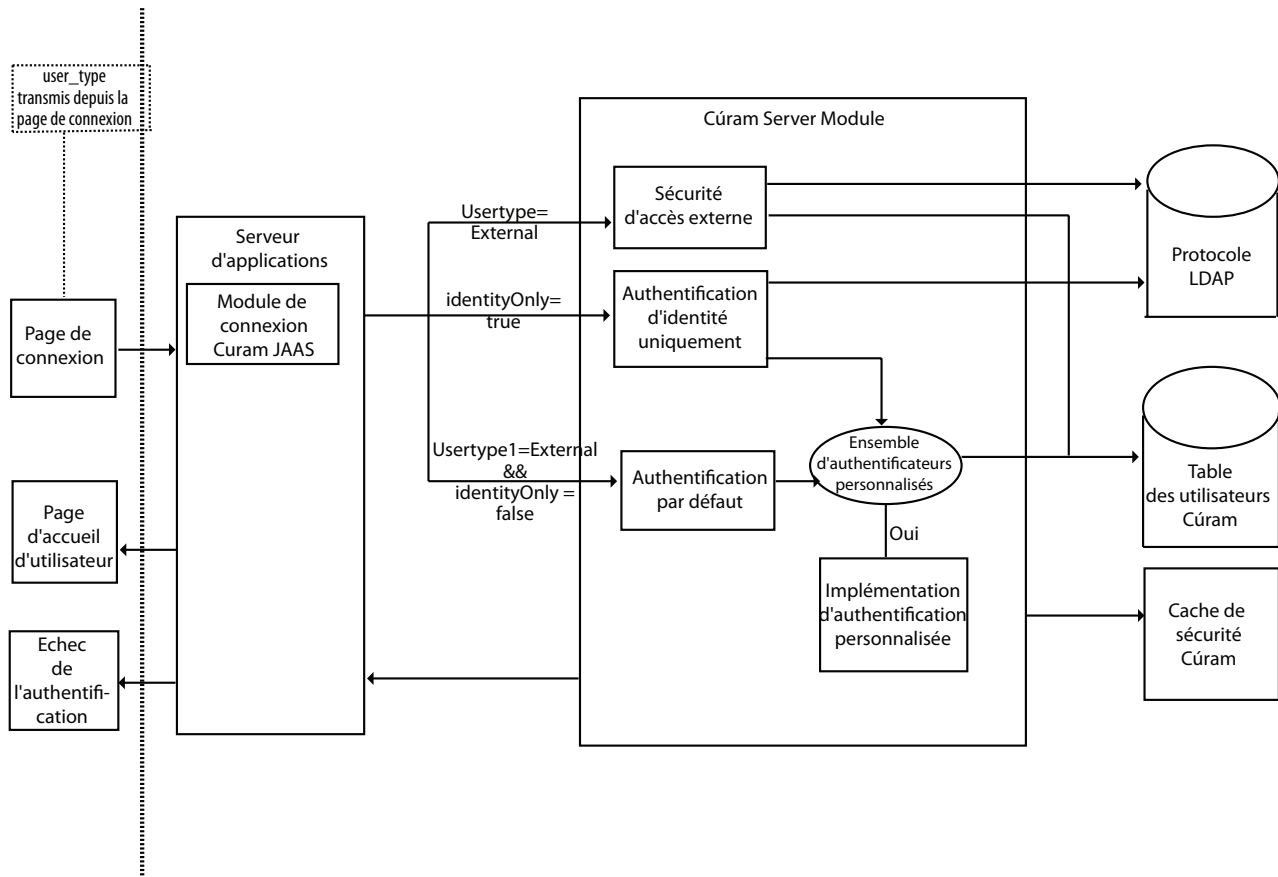


Figure 1. Architecture d'authentification

La figure 2.3, «Architecture d'authentification» ci-dessus présente l'architecture du processus d'authentification d'un utilisateur. Prête à l'emploi, l'authentification par défaut est effectuée pour un utilisateur. Ce comportement peut être personnalisé pour les utilisateurs internes et externes, selon les exigences d'authentification. Les sections suivantes de ce chapitre décrivent en détail chacune des zones fonctionnelles qui composent l'architecture d'authentification en indiquant où des personnalisations sont possibles.

2.4 Authentification par défaut

L'authentification par défaut prête à l'emploi pour IBM Cúram Social Program Management nécessite que l'utilisateur se connecte via l'écran de connexion, où l'utilisateur est invité à entrer un nom d'utilisateur et un mot de passe comme données d'identification. Ces données d'identification sont ensuite transmises au module de connexion JAAS Cúram configuré dans le serveur d'applications.

L'authentification par défaut est appelée et le nom d'utilisateur et le mot de passe entrés sont vérifiés par rapport au nom d'utilisateur et au mot de passe stockés sur la table de base de données des utilisateurs Cúram. Le nom d'utilisateur Cúram n'est pas modifiable, mais vous pouvez configurer votre système pour utiliser à la place un ID de connexion Cúram, qui est modifiable. L'ID de connexion est une extension logique de l'utilisateur Cúram. Les mêmes vérifications contrôlées au niveau du nom d'utilisateur sont également contrôlées au niveau de l'ID de connexion. Voir 2.5, «Autres ID de connexion», à la page 5 pour plus d'informations sur les autres ID de connexion.

L'authentification effectue plusieurs vérifications au niveau des données d'identification de connexion. Voir 2.14, «Authentification par défaut», à la page 11 pour plus d'informations sur les vérifications.

Si toutes les vérifications sont réussies, l'utilisateur est considéré comme authentifié par l'application.

Une fois que l'utilisateur est authentifié, il peut ensuite être ajouté au cache de sécurité Cúram. Le cache de sécurité Cúram stocke toutes les données d'autorisation connexes pour cet utilisateur afin d'optimiser l'extraction de données d'autorisation pour un utilisateur. Pour plus d'informations sur le cache de sécurité Cúram, consultez la rubrique Chapitre 5, «Mise en cache des données de sécurité», à la page 23. La figure 2.3 ci-dessous met en évidence le chemin emprunté par l'authentification par défaut.

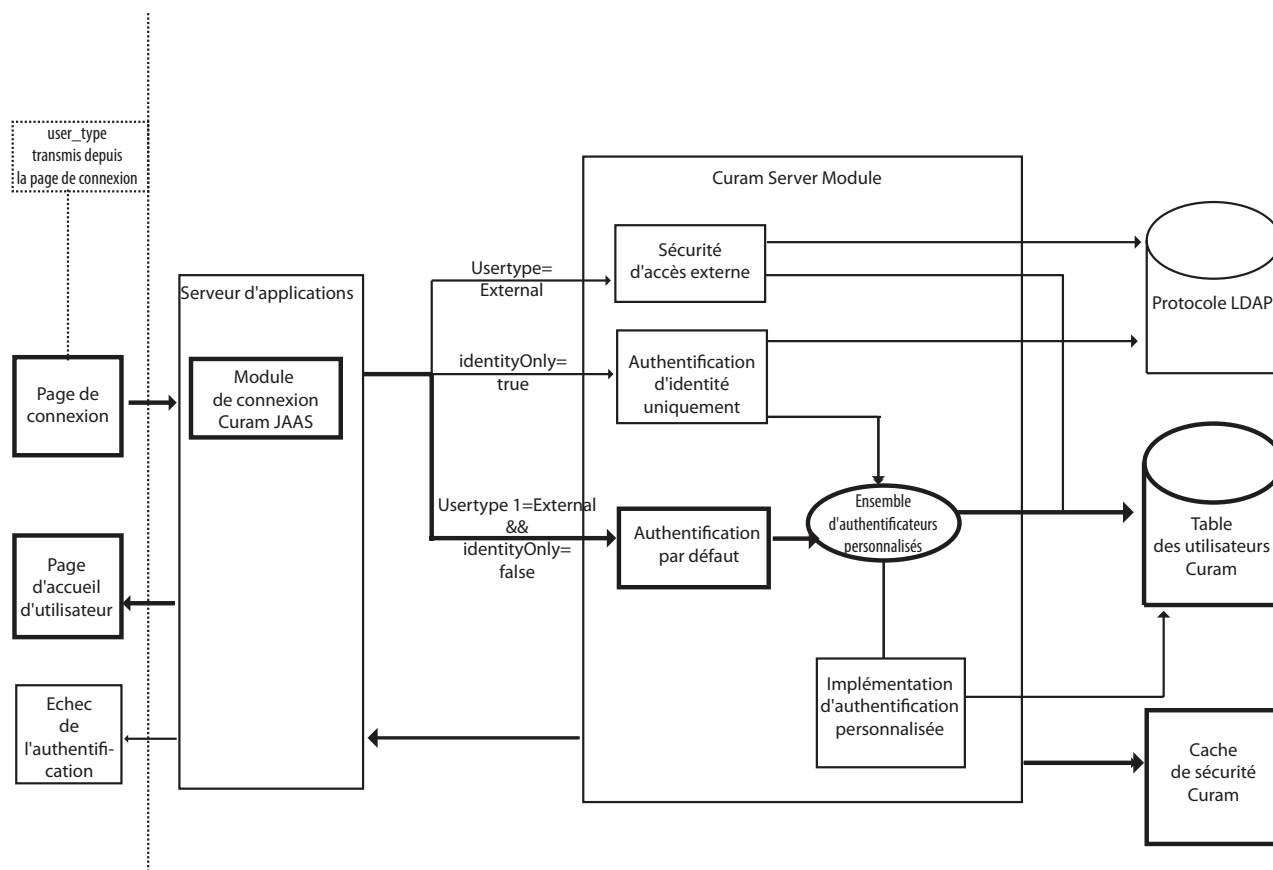


Figure 2. Authentification par défaut

2.5 Autres ID de connexion

Par défaut, Cúram utilise le nom d'utilisateur et mot de page traité stockés dans la table Utilisateurs dans le cadre de l'authentification. Une fois créé, ce nom d'utilisateur n'est pas modifiable. Ce manque de flexibilité peut ne pas correspondre aux exigences de certaines installations. Toutefois, cet ID de connexion peut être remplacé par un ID de connexion qui peut être mis à jour. L'ID de connexion fonctionne comme une extension logique de la tables d'utilisateurs Cúram. Lorsque l'autre ID de connexion est utilisé, le nom d'utilisateur existe toujours et est utilisé en interne par Cúram. Toutefois, l'utilisateur se connecte à Cúram à l'aide de l'ID de connexion.

Éléments à prendre en compte lors de l'utilisation de l'autre ID de connexion :

- L'autre ID de connexion et le nom d'utilisateur s'excluent mutuellement. Cela signifie que les utilisateurs ne peuvent pas se connecter avec des noms d'utilisateur et des ID de connexion.

- La table `Cúram ExtendedUsersInfo` qui contient l'ID de connexion stocké doit être renseignée avant d'activer la fonction de l'autre ID de connexion, qui est expliquée plus en détail ci-dessous.
- Lors de l'utilisation d'ID de connexion, les résultats d'authentification sont stockés dans la table `AuthenticationLog` et la colonne `AltLogin` indique si la colonne `UserName` représente un nom d'utilisateur (false) ou un ID de connexion (true).
- Les ID de connexion ne s'appliquent qu'aux utilisateurs Cúram internes (par ex., utilisateurs stockés dans la table d'utilisateurs Cúram). Toutefois, si vous utilisez l'authentification d'identité seule avec d'autres ID de connexion, ces ID doivent correspondre aux ID de connexion stockés dans la table `Cúram ExtendedUsersInfo` (registre WebSphere, protocole LDAP, etc.).
- Lors de l'affectation des ID de connexion, vous devez vous charger des ID qui sont utilisés en interne et/ou ont des dépendances (par ex., avec des valeurs de propriété) en dehors de la table des utilisateurs Cúram. Il s'agit des noms d'utilisateur qui génèrent des problèmes si son ID de connexion diffère du nom d'utilisateur sans changement correspondant, comme indiqué :
 - `SYSTEM` - Dans WebSphere, ce nom d'utilisateur est associé au traitement JMS et est inclus à la configuration de WebSphere au moment de la phase de déploiement de l'application. Voir 6.2, «Utilisateurs Cúram obligatoires», à la page 25 et le manuel *WebSphere Cúram - Guide de déploiement* pour plus d'informations sur la changement de cet ID.
 - `DBTOJMS` - Il s'agit du nom d'utilisateur `DBtoJMS` par défaut utilisé par le traitement par lots et référencé par la propriété `curam.security.credentials.dbtojms.username`. Voir 6.2, «Utilisateurs Cúram obligatoires», à la page 25, 6.5, «Messagerie JMS», à la page 26, 6.6, «Traitement différé», à la page 27 et le manuel *Cúram - Guide de traitement par lots* pour plus d'informations.
 - `WEBSVCS` - Il s'agit du nom d'utilisateur des services Web par défaut référencé par la propriété `curam.security.credentials.dbtojms.username`. Voir 6.2, «Utilisateurs Cúram obligatoires», à la page 25, 6.3, «Services Web», à la page 25 et le manuel *Cúram - Guide des services Web* pour plus d'informations.
 - `unauthenticated` - Donnée principale qu'utilise WebSphere pour les utilisateurs non authentifiés. Cet ID de connexion ne doit pas être changé.

Pour pouvoir utiliser l'autre ID de connexion une fois la table `ExtendedUsersInfo` renseignée, définissez la propriété `curam.security.altlogin.enabled` sur `true` (voir le *Guide de développement du serveur Cúram* pour plus d'informations sur les propriétés Cúram). Il s'agit d'une propriété statique. Vous devez redémarrer Cúram pour qu'elle devienne effective.

Pour renseigner la table `ExtendedUsersInfo` avant d'activer la fonction, plusieurs options sont disponibles :

- Une simple instruction SQL suffit pour renseigner la table à l'aide du nom d'utilisateur dans la table `Utilisateurs` ; par conséquent, il n'y a pas d'impact immédiat au niveau des utilisateurs : `INSERT INTO EXTENDEDUSERSINFO (USERNAME, LOGINID, UPPERLOGINID, VERSIONNO) (SELECT USERNAME, USERNAME, UPPER(USERNAME), 1 FROM USERS)` ; Vous pouvez ensuite déployer vos modifications aux ID de connexion de manière contrôlée.
- Vous pouvez implémenter une application SQL pour le mappage de votre nom d'utilisateur et de votre ID de connexion (par ex., noms usuels de protocole LDAP).

Remarque : Vous devez maintenir la relation de clé externe du nom d'utilisateur entre les tables `Users` et `ExtendedUsersInfo`.

2.6 Page de connexion

La page de connexion prête à l'emploi par défaut est représentée par le fichier `logon.jsp`. Ce fichier `logon.jsp` représente la page de connexion permettant à l'utilisateur de procéder à l'authentification de la connexion par formulaire. Par défaut, le fichier `logon.jsp` contient les zones de nom d'utilisateur et de mot de passe. Toutefois, le fichier `logon.jsp` peut être personnalisé pour transmettre un paramètre supplémentaire en ajoutant la zone de type d'utilisateur. Cette zone détermine le type d'utilisateur en

cours de connexion, c'est-à-dire utilisateur interne ou externe. Le nom d'utilisateur, le mot de passe et le type d'utilisateur (s'il existe) sont tous transmis au module de connexion JAAS Cúram dans le cadre du processus d'authentification.

Le fichier `logon.jsp` prêt à l'emploi par défaut ne dispose pas de la propriété `user_type` définie. Si cette propriété est omise, l'utilisateur est censé être interne. Lorsque cette propriété est définie, cela indique qu'un utilisateur externe est en train de se connecter. Cette propriété peut être définie sur n'importe quelle valeur autre que 'INTERNE'.

2.7 Personnalisation de la page de connexion

Le fichier `logon.jsp` peut être personnalisé, c'est-à-dire que le fichier `logon.jsp` peut être complètement remplacé par un fichier `logon.jsp` personnalisé, pour les raisons suivantes notamment :

Une application client d'utilisateur externe est en cours de développement ;

Si une application client d'utilisateur externe est en cours de développement, un nouveau fichier `logon.jsp` doit être créé, car le type d'utilisateur doit être défini de manière à indiquer qu'un utilisateur externe est en train de se connecter. Pour plus d'informations, reportez-vous à la rubrique 13.3, «Création d'une page de connexion client d'utilisateur externe», à la page 51.

La connexion automatique est nécessaire ;

Certaines applications client d'utilisateur externe ne nécessitent aucune authentification d'utilisateur, aucun nom d'utilisateur ni mot de passe n'est donc demandé (notamment dans le cas d'une application d'accès public externe). Il est impossible de désactiver l'authentification, le meilleur moyen de satisfaire cette exigence consiste donc à écrire un script de connexion automatique. Pour ce faire, le fichier `logon.jsp` doit être personnalisé pour l'application d'accès public externe. Pour plus d'informations, reportez-vous à la rubrique 13.4, «Création d'une page de connexion automatique de client de l'utilisateur externe», à la page 51.

Un style différent est obligatoire ;

Consultez la section sur les pages de connexion du *Manuel de référence du client Web Cúram* pour plus d'informations sur le style concernant le fichier `logon.jsp`.

Il existe une exigence concernant les noms d'utilisateur qui doivent contenir des caractères étendus (valide uniquement pour Oracle WebLogic Server).

WebLogic Server fournit un attribut de propriété, `j_character_encoding`, qu'il doit être ajouté au fichier `logon.jsp`. Pour plus d'informations, consultez la rubrique 10.3, «Activation des noms d'utilisateurs avec caractères étendus pour WebLogic Server», à la page 37.

2.8 Module de connexion JAAS Cúram

L'authentification est effectuée par un module de connexion JAAS. Ce dernier est configuré dans le serveur d'applications et est appelé automatiquement par le serveur d'applications dans le cadre du processus d'authentification pour tout accès à l'application IBM Cúram Social Program Management. L'avantage de cette approche est que le mode d'authentification par défaut peut être utilisé avec ou remplacé par une approche personnalisée, sans affecter l'application IBM Cúram Social Program Management.

Comme indiqué précédemment, le module de connexion JAAS Cúram peut être configuré pour fonctionner selon trois modes. Pour plus d'informations sur la configuration des modules de connexion et sur un comportement spécifique du serveur d'applications, consultez la section sur la configuration du serveur d'applications dans le *Guide de déploiement de serveur Cúram* du serveur d'applications utilisé.

Les exigences spécifiques au projet peuvent signifier que plusieurs modules de connexion sont nécessaires. Par exemple, un utilisateur peut avoir à entrer des éléments en plus du nom d'utilisateur et du mot de passe à des fins de vérification. Il est possible de configurer plusieurs modules de connexion dans le serveur d'applications. Chaque module de connexion est exécuté dans l'ordre déterminé par les paramètres dans le serveur d'applications. Pour plus d'informations sur ces paramètres, consultez la documentation WebSphere ou WebLogic Server.

Une fois que l'utilisateur est bien authentifié par tous les modules de connexion nécessitant une authentification réussie de l'utilisateur (celle-ci est configurable dans le serveur d'applications), il est considéré comme authentifié par l'application.

2.9 Gestion des mots de passe

Les mots de passe de tous les utilisateurs Cúram internes et externes sont stockés sous leur forme chiffrée dans les tables de bases de données d'utilisateurs et d'utilisateurs externes Cúram. Lorsque le module de connexion JAAS Cúram reçoit le mot de passe, il est chiffré avant d'être envoyé au bean de connexion pour comparaison. Ce traitement est un algorithme unidirectionnel permettant de garantir la sécurité du mot de passe. Le mot de passe stocké pour l'utilisateur sur la base de données utilise le même algorithme de traitement, basé sur vos paramètres de chiffrement, garantissant ainsi la comparaison correcte des mots de passe chiffrés, ainsi que la sécurité.

Les utilisateurs gérés en externe (par ex., via protocole LDAP avec authentification d'identité seule) ne sont pas concernés par le processus ci-dessus. Lors de l'authentification par rapport à un système tiers (par ex., serveur LDAP ou SSO), où l'application Cúram doit transmettre les données d'identification saisies par l'utilisateur, l'implémentation personnalisée de `curam.util.security.PublicAccessUser` peut être utilisée, car elle permet d'accéder aux données d'identification via un mot de passe en texte clair.

2.10 Configuration par défaut de WebLogic Server

Le module de connexion JAAS est configuré en tant que fournisseur d'authentification WebLogic Server . Le fournisseur d'authentification est le seul fournisseur configuré par les scripts de configuration fournis par WebLogic Server . Comme il s'agit du seul fournisseur d'authentification configuré, le fournisseur d'authentification Cúram est responsable de l'authentification et de la vérification de l'utilisateur. Comme indiqué au préalable, il se peut que plusieurs fournisseurs d'authentification soient configurés dans WebLogic Server. Dans ce cas, le fournisseur d'authentification Cúram peut ne pas être responsable de l'authentification et de la vérification de l'utilisateur. Pour plus d'informations, consultez la rubrique 8.3, «Connexion unique avec WebLogic Server», à la page 34.

2.11 Configuration par défaut de WebSphere

Le module de connexion JAAS Cúram est configuré en tant que système WebSphere . La valeur par défaut, la configuration des paramètres de sécurité scriptée dans WebSphere, nécessite le registre d'utilisateurs basés sur les fichiers par défaut et le module de connexion système Cúram. Le registre d'utilisateurs dans WebSphere correspond au mode d'authentification par défaut et peut être configuré en tant que :

- Registre d'utilisateurs personnalisé ;
- Serveur d'annuaire LDAP ;
- Système d'exploitation local ou ;
- Référentiel de fichiers WebSphere.

Il existe plusieurs configurations de connexion système de WebSphere . Le module de connexion système Cúram est configuré pour les configurations DEFAULT, WEB_INBOUND et RMI_INBOUND. Le même module de connexion est utilisé pour les trois configurations. WebSphere appelle automatiquement les modules de connexion configurés en tant que modules de connexion système sous certaines circonstances :

- DEFAULT

Les modules de connexion spécifiés pour la configuration DEFAULT sont appelés pour l'authentification des services Web et des appels JMS. Ils sont également appelés lors de la phase de démarrage de WebSphere.

- WEB_INBOUND

Les modules de connexion spécifiés pour la configuration WEB_INBOUND sont utilisés pour l'authentification des requêtes Web.

- RMI_INBOUND

Les modules de connexion spécifiés pour la configuration RMI_INBOUND sont utilisés pour l'authentification des clients Java.

Le module de connexion JAAS Cúram existe comme module de connexion dans une chaîne de modules de connexion configurés dans WebSphere . Au moins un de ces modules de connexion doit ajouter des données d'identification pour l'utilisateur. Par défaut, le module de connexion Cúram ajoute des données d'identification pour un utilisateur authentifié. Par conséquent, le registre d'utilisateurs WebSphere configuré géré par un module de connexion ultérieur n'ajoute pas de données d'identification. Il n'est donc pas nécessaire de définir les utilisateurs Cúram dans le registre d'utilisateurs WebSphere. Ce comportement est configurable à l'aide de la propriété `curam.security.user.registry.enabled` définie dans le fichier `AppServer.properties`. Pour plus d'informations sur la définition de cette propriété, consultez le *Guide de déploiement Cúram pour WebSphere Application Server* ou le *Guide de déploiement Cúram pour WebSphere Application Server on z/OS*. La rubrique 2.11, «Configuration par défaut de WebSphere», à la page 8 ci-dessous présente le flux d'authentification par défaut de WebSphere . La rubrique 2.11, «Configuration par défaut de WebSphere», à la page 8 ci-dessous présente le flux d'authentification WebSphere où le registre d'utilisateurs associé est également requis, c'est-à-dire où la propriété `curam.security.user.registry.enabled` est définie sur `true`.

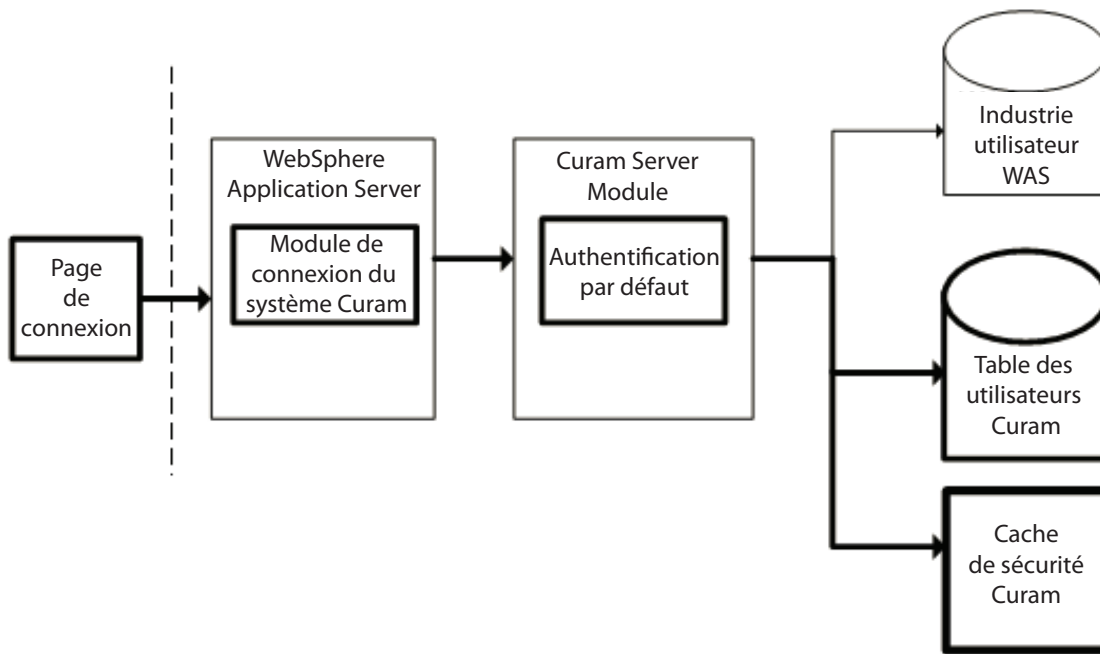


Figure 3. Flux d'authentification par défaut de WebSphere

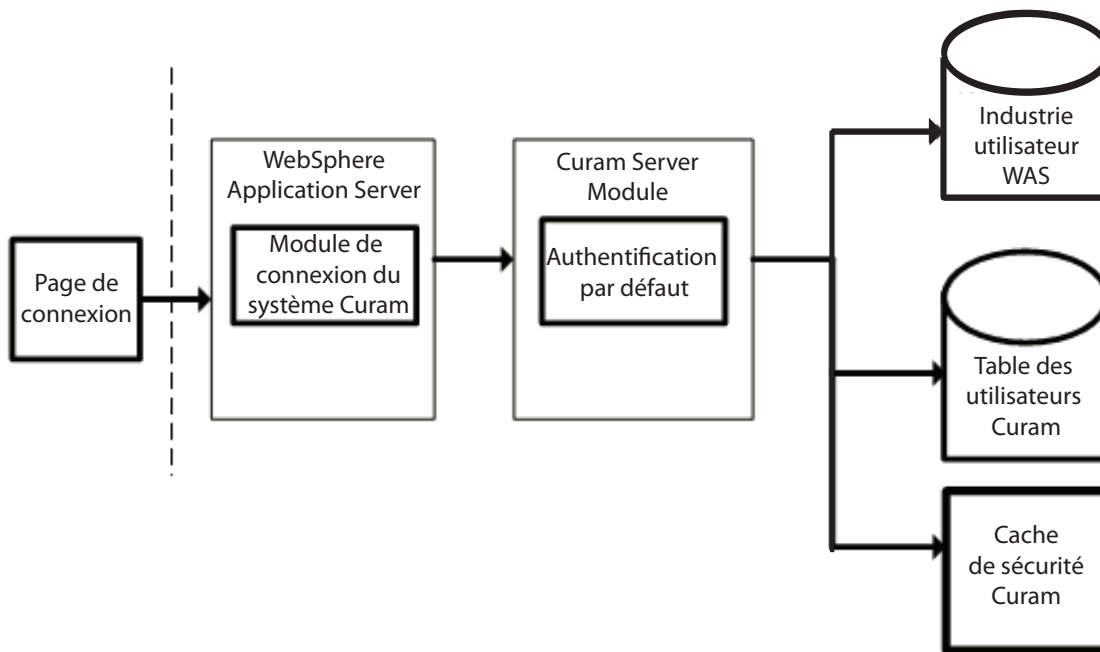


Figure 4. Flux d'authentification de WebSphere avec le registre d'utilisateurs activé

Dans le cadre de la configuration des paramètres de sécurité, certains utilisateurs sont exclus de l'authentification et, pour ces derniers, le registre d'utilisateurs configurés *est* requis. Les utilisateurs de cette liste sont configurés automatiquement en tant qu'utilisateurs de sécurité WebSphere, comme indiqué par la propriété `security.username` dans le fichier `AppServer.properties` et en tant qu'utilisateurs de base de données, comme indiqué par la propriété `curam.db.username` dans le fichier `Bootstrap.properties`. Ces deux utilisateurs sont classés comme administrateurs et non comme utilisateurs d'application. Il est possible d'étendre manuellement cette liste d'utilisateurs exclus. Pour plus d'informations, consultez le *Guide de déploiement Cúram pour WebSphere Application Server* et le *Guide de déploiement Cúram pour WebSphere Application Server on z/OS*.

Avertissement : Les utilisateurs `security.username` et `curam.db.username` sont automatiquement ajoutés au référentiel d'utilisateurs WebSphere basé sur les fichiers WebSphere par les scripts de configuration fournis. Si le registre d'utilisateurs WebSphere configuré n'est pas le registre par défaut, ces utilisateurs doivent exister dans l'autre registre d'utilisateurs WebSphere.

2.12 Personnalisation du module de connexion JAAS

Il se peut que le module de connexion JAAS Cúram ne prenne pas en charge les exigences d'authentification pour une solution personnalisée en particulier. Nous recommandons vivement que, lors du développement d'un module de connexion personnalisé, le module de connexion JAAS Cúram reste en place et soit utilisé avec l'authentification d'identité uniquement activée. Cependant, si nécessaire, le module de connexion JAAS Cúram peut être supprimé et remplacé par une solution personnalisée. Dans ce cas, vous devez consulter le support.

Avertissement : S'il est possible de supprimer complètement le module de connexion JAAS Cúram, les utilisateurs doivent toujours exister dans la table de base de données des utilisateurs Cúram pour des motifs d'autorisation.

Le module de connexion JAAS Cúram ajoute de nouveaux utilisateurs au cache de sécurité Cúram automatiquement, et lorsque ce module de connexion JAAS Cúram est remplacé par un module de connexion JAAS personnalisé, cette fonctionnalité n'est plus présente. Si un module de connexion JAAS personnalisé remplace complètement le module de connexion JAAS Cúram, le module de connexion JAAS personnalisé doit s'assurer qu'une mise à jour du cache de sécurité est déclenchée dès qu'un nouvel utilisateur est ajouté à la base de données.

2.13 Processus de vérification pour l'authentification

Le type des vérifications effectuées dépend du mode d'authentification utilisé. Vous trouverez ci-dessous une liste des modes/configurations d'authentification ainsi que les détails complets sur les vérifications réalisées pour chaque mode d'authentification.

2.14 Authentification par défaut

L'authentification par défaut fait partie de la configuration prête à l'emploi et ce mode d'authentification implique la vérification du nom d'utilisateur et du mot de passe spécifiés par rapport à la table de base de données des utilisateurs Cúram. Toutes les informations de connexion de ce dossier sont conservées par l'application IBM Cúram Social Program Management.

2.15 Processus de vérification par défaut

Les vérifications effectuées par le module de connexion Cúram lors de l'authentification par défaut sont les suivantes :

- nom d'utilisateur et mot de passe
- expiration de compte et/ou de mot de passe
- synchronisation de nom d'utilisateur avec le cache de sécurité

- détection d'intrusion, par ex. nombre maximum de tentatives de saisie de mot de passe, noms d'utilisateurs incorrects, échecs de changement de mot de passe
- restrictions d'accès par jour et par heure - jour de la semaine et plage horaire dans la journée

L'authentification et l'autorisation des noms d'utilisateurs sont sensibles à la casse par défaut, toutefois, il est possible de désactiver l'authentification sensible à la casse. S'il existe des doublons de noms d'utilisateurs insensibles à la casse (par exemple responsable du dossier, Responsable du Dossier), l'authentification échoue en raison d'un nom d'utilisateur ambigu. Pour plus d'informations, consultez la rubrique 10.4, «Changement de la sensibilité à la casse du nom d'utilisateur», à la page 37.

2.16 Tentatives d'authentification

Les échecs d'authentification ne sont pas rapportés directement à un client car cela fournirait des informations supplémentaires à un intrus essayant de s'introduire dans le système. Par exemple, le signalement d'un mot de passe incorrect pourrait indiquer que le nom d'utilisateur est valide. Toutes les tentatives d'authentification (qu'elles aboutissent ou échouent) sont consignées dans une table de base de données appelée AuthenticationLog. Pour plus d'informations, consultez la rubrique 11.15, «Analyse de la table de base de données AuthorisationLog», à la page 44.

2.17 Personnalisation de l'authentification par défaut

L'implémentation par défaut peut être personnalisée pour utiliser un ID de connexion modifiable au lieu du nom d'utilisateur Curam et la possibilité d'ajouter des vérifications supplémentaires en implémentant l'authentificateur personnalisé (2.21, «Vérifications personnalisées», à la page 14 doit être référencée pour plus d'informations).

2.18 Authentification d'identité uniquement

L'authentification peut être configurée pour effectuer une vérification d'identité uniquement, au lieu des vérifications par défaut répertoriées dans la rubrique 2.15, «Processus de vérification par défaut», à la page 11 ci-dessus.

La vérification d'identité uniquement signifie que le mode d'authentification garantit seulement que le nom d'utilisateur de l'utilisateur qui se connecte existe sur la table de base de données des utilisateurs Cúram. L'authentification complète doit être effectuée par un autre mode, afin d'être configurée dans le serveur d'applications.

Un autre mode peut être, par exemple, un serveur d'annuaire LDAP, qui est pris en charge en tant que mode d'authentification à la fois par les serveurs d'applications WebSphere et WebLogic Server. Une autre possibilité consiste à utiliser une solution de connexion unique pour l'authentification, ou d'implémenter un module de connexion personnalisé. Pour découvrir les solutions de serveur d'applications personnalisées, consultez la documentation d'IBM ou d'Oracle.

Avec l'authentification d'identité uniquement (comme pour l'authentification par défaut), les entrées sont ajoutées à la table de base de données AuthenticationLog à la fin du processus d'authentification.

Pour une connexion réussie, le statut suivant est utilisé :

- AUTHONLY

Pour un scénario d'échec, le statut suivant est utilisé :

- BADUSER

Il s'agit du seul scénario d'échec possible où il n'existe pas d'utilisateur.

Les zones loginFailures et lastLogin du fichier AuthenticationLog ne sont pas définies. Cela est vrai même si les vérifications personnalisées sont implémentées.

Lorsque les informations d'expiration du mot de passe pour un utilisateur sont définies (sur la table de base de données des utilisateurs Cúram), l'avertissement d'expiration du mot de passe s'affiche si celui-ci arrive à expiration. Avec l'authentification d'identité uniquement, cet avertissement est trompeur. Il est recommandé que les zones relatives aux vérifications d'authentification, comme l'expiration du mot de passe ou le compte activé, ne soient pas utilisées si l'authentification d'identité uniquement est activée.

Lorsque l'authentification d'identité uniquement est activée, la sécurité n'est pas utilisée pour l'authentification mais elle l'est toujours à des fins d'autorisation. Par conséquent, tous les utilisateurs qui ont besoin d'accéder à l'application doivent toujours exister dans la table de base de données des utilisateurs Cúram, ainsi que dans l'autre mode d'authentification, par exemple LDAP. Il est important de noter que deux utilisateurs doivent exister dans les deux emplacements à la fois, à savoir l'utilisateur SYSTEM et l'utilisateur DBTOJMS. Pour plus d'informations sur ces utilisateurs, consultez la rubrique Chapitre 6, «Sécurité des autres clients», à la page 25.

Pour plus d'informations sur la méthode de configuration de l'identité uniquement pour un serveur d'applications, consultez la rubrique 10.7, «Configuration de l'authentification d'identité uniquement», à la page 38.

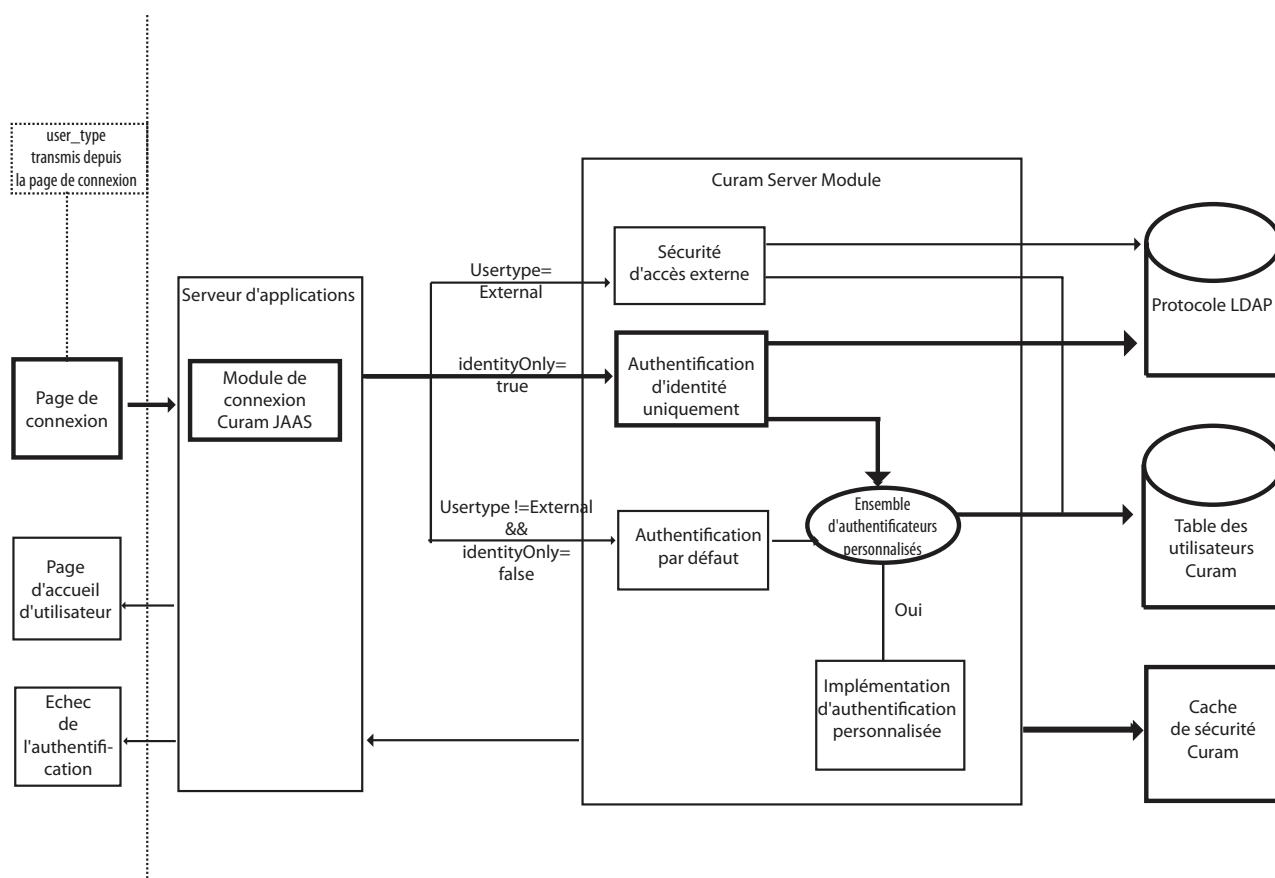


Figure 5. Authentification d'identité uniquement

2.19 Personnalisation de l'authentification d'identité uniquement

L'implémentation de l'identité uniquement n'est pas personnalisable mais il est possible d'ajouter des vérifications supplémentaires en implémentant l'authentificateur personnalisé. Pour plus d'informations, consultez la rubrique 2.21, «Vérifications personnalisées», à la page 14.

2.20 Authentification de sécurité d'accès externe

L'architecture permet à un développeur d'implémenter sa propre solution d'authentification personnalisée pour les utilisateurs externes en fournissant un «point d'ancrage» dans l'infrastructure d'authentification et d'autorisation existante de l'environnement CDEJ.

Pour «ancrer» la solution personnalisée dans l'application, la classe `curam.util.security.PublicAccessUser` doit être étendue. Cela nécessite d'implémenter l'interface `curam.util.security.ExternalAccessSecurity`. Cette classe est utilisée lors des processus d'authentification et d'autorisation pour déterminer les informations requises relatives à l'utilisateur externe. Pour plus d'informations, consultez la rubrique Chapitre 13, «Personnalisation des applications d'utilisateurs externes», à la page 51.

2.21 Vérifications personnalisées

La prise en charge est fournie pour l'ajout de vérifications personnalisées au processus d'authentification. Par exemple, un utilisateur peut devoir répondre à une question de sécurité qui doit ensuite être vérifiée. Le code personnalisé, s'il est implémenté, est appelé après les vérifications IBM Cúram Social Program Management pertinentes ou la vérification d'identité, et uniquement si celles-ci ont abouti.

Après que les vérifications personnalisées ont été appelées, le processus d'authentification met à jour les zones appropriées sur la table de base de données des utilisateurs.

Pour plus d'informations, consultez la rubrique 10.5, «Ajout de vérifications personnalisées au processus d'authentification», à la page 37.

Chapitre 3. Autorisation

3.1 Présentation

Dans IBM Cúram Social Program Management, le processus d'octroi ou de refus de l'accès des utilisateurs aux éléments fonctionnels d'une application est appelé autorisation. L'élément fonctionnel peut être tout élément auquel il est possible d'attacher un identificateur unique, notamment :

- un appel de processus serveur,
- un élément de l'application qui nécessite une vérification de la sécurité, par exemple une gamme de produits de bien-être déposés.

L'accès à l'élément fonctionnel est contrôlé par un identificateur de sécurité (SID) faisant partie des données d'autorisation d'IBM Cúram Social Program Management. Ces données sont reliées à un utilisateur et peuvent être configurées via les écrans Cúram Administration ou le gestionnaire de données. Pour plus d'informations, consultez le *Guide de développement de serveur Cúram*.

Les données de sécurité créées pour l'autorisation sont centralisées dans le traitement effectué lors de chaque appel client-serveur, et il est important que l'accès soit optimisé pour des raisons de performances. Le cache de sécurité Cúram est responsable de la mise en cache des données d'autorisation pour un utilisateur. Pour plus d'informations, consultez la rubrique 5.2, «Cache de sécurité Cúram», à la page 23.

Les sections suivantes décrivent la relation entre ces concepts d'autorisation et leur mode de fonctionnement dans IBM Cúram Social Program Management.

3.2 Utilisateurs, rôles et groupes

Les informations de sécurité associées à une application doivent d'abord être organisées dans des profils de sécurité avant de pouvoir être utilisées dans un environnement d'exécution. Un profil de sécurité comprend un rôle de sécurité, un ou plusieurs groupes de sécurité ainsi que les associations entre les identificateurs de sécurité et les éléments sécurisables d'une application.

Chaque utilisateur autorisé est affecté à un rôle de sécurité lors de la configuration des paramètres de sécurité et les rôles ainsi constitués sont associés à un certain nombre de groupes de sécurité. Chaque groupe de sécurité est associé à un certain nombre d'identificateurs de sécurité. L'identificateur de sécurité représente les éléments sécurisables d'IBM Cúram Social Program Management, comme une méthode ou un champ. Les informations sur le rôle, les groupes et l'identificateur sont stockées sur la base de données dans un certain nombre de tables et sont configurées à l'aide du gestionnaire de données d'application ou des écrans Cúram Administration.

Cette structure de données permet d'autoriser tous les utilisateurs par rapport à un tout élément sécurisé d'une application. Il s'agit d'une méthode efficace et souple à la fois permettant d'octroyer une autorisation aux utilisateurs Cúram.

Un ensemble minimum d'identificateurs de sécurité est obligatoire pour qu'un utilisateur puisse exploiter l'application Social Program Management Platform. Ces identificateurs de sécurité sont associés au groupe prêt à l'emploi BASESECURITYGROUP. Pour connaître la liste de ces identificateurs de sécurité, consultez le fichier `EJBServer/components/core/data/initial/handcraftedscripts/Supergroup.sql`. Ce fichier vise à relier les identificateurs de sécurité au groupe prêt à l'emploi BASESECURITYGROUP.

Une manière simple de s'assurer que tous les utilisateurs possèdent les droits relatifs à cet ensemble d'identificateurs de sécurité consiste à créer un groupe de sécurité unique pour ceux-ci puis à associer ce groupe de sécurité à chaque rôle de sécurité dans le système.

3.3 Identificateurs de sécurité

Chaque élément sécurisé dans IBM Cúram Social Program Management se voit attribuer un identificateur de sécurité (SID) unique à travers toute l'application.

Le processus d'autorisation est généré dans l'infrastructure et, une fois que les éléments sécurisés ont été identifiés, les autres éléments sont gérés par les générateurs de code, les scripts et les écrans Cúram Administration. L'analyse des éléments à sécuriser est un processus manuel qui doit être effectué par le développeur ou un administrateur de sécurité. Cette section présente l'infrastructure disponible pour configurer l'autorisation.

Le premier type d'autorisation à prendre en compte est celui de la méthode (façade) de processus, également connu sous le nom de *sécurité fonctionnelle*. Dans le modèle Cúram, un développeur peut décider si la sécurité est activée ou désactivée au niveau de la méthode de processus. L'option s'applique uniquement aux objets de processus métier car ces derniers contiennent les appels exposés au client. Les méthodes d'objet d'entité ne sont pas incluses dans le processus d'autorisation.

Il existe plusieurs types d'identificateurs de sécurité, notamment :

- Identificateurs de fonction
- Identificateurs de sécurité de niveau zone
- Types d'identificateurs de sécurité définis par l'utilisateur.

3.4 Identificateurs de fonction

Les identificateurs de fonction sont un type spécifique d'identificateur de sécurité où le type est défini sur FUNCTION. Lorsqu'une méthode est mise à la disposition du public (en définissant le stéréotype en tant que façade dans le modèle), un identificateur de fonction est généré pour cette méthode et la sécurité est automatiquement activée.

Il est possible de désactiver la sécurité pour une méthode de processus lors de la phase de conception. Pour plus d'informations, consultez la rubrique 11.11, «Désactivation de la sécurité pour une méthode de processus», à la page 43.

3.5 Identificateurs de sécurité de niveau zone

Un identificateur de sécurité de niveau zone permet d'appliquer l'autorisation à des zones spécifiques sur une méthode accessible au public. Lors de l'exécution, si un utilisateur ne dispose pas des droits d'accès pour visualiser la zone à afficher, les contenus de la zone sont affichés sous forme de plusieurs astérisques (***) . Pour plus d'informations sur les identificateurs de sécurité de niveau zone, consultez le *Guide de référence de modélisation Cúram*.

3.6 Identificateurs de sécurité définis par l'utilisateur

Dans les sections précédentes, nous avons décrit

les identificateurs de fonction ;

Il s'agit d'un identificateur de sécurité généré automatiquement de type fonction.

les identificateurs de sécurité de niveau zone ;

Il s'agit de la sécurité appliquée à des zones spécifiques sur une méthode.

Il existe également le concept d'identificateur de sécurité défini par l'utilisateur. Le processus d'autorisation est suffisamment souple pour s'adapter à n'importe quel élément sécurisable d'une application IBM Cúram Social Program Management. Le développeur peut personnaliser efficacement le processus d'autorisation en définissant de nouveaux *types* d'identificateurs de sécurité. Les nouveaux types représentent un élément conceptuel nécessitant la sécurité. La méthode d'interface de serveur suivante permet d'appeler l'autorisation directement sur ces nouveaux types d'identificateurs de sécurité définis par l'utilisateur.

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

Prêts à l'emploi, les identificateurs de sécurité LOCATION et PRODUCT sont des identificateurs de sécurité de ce type. Avec la méthode ci-dessus, il n'existe effectivement aucune limite aux types d'identificateurs de sécurité qui peuvent être définis. Pour plus d'informations, consultez la rubrique 11.14, «Autorisation de nouveaux types d'identificateurs de sécurité», à la page 44.

3.7 Autorisation d'exécution

L'infrastructure IBM Cúram Social Program Management effectue des vérifications d'autorisation au niveau du client Web et du serveur à la fois.

3.8 Vérifications d'autorisation client

Pour qu'un utilisateur puisse accéder à une méthode ou à une zone, le client Web effectue des vérifications d'autorisation avant le chargement initial de la page. Si l'utilisateur ne dispose d'aucun accès, la vérification d'autorisation client échoue et le serveur n'est pas appelé. Cette vérification est configurable dans le fichier `curam-config.xml` en définissant la propriété `SECURITY_CHECK_ON_PAGE_LOAD`. Pour plus d'informations, consultez la section 3.12.13, Configuration générale du *Manuel de référence du client Web Cúram*.

Par défaut, les échecs de vérification d'autorisation de client Web ne sont pas enregistrés. Ce comportement est configurable. Pour plus d'informations, consultez la rubrique 11.13, «Contrôle de la consignation des échecs d'autorisation du client», à la page 43.

3.9 Vérifications d'autorisation de serveur

Pour gérer les autres accès à IBM Cúram Social Program Management et lorsque la vérification d'autorisation de client Web est désactivée, une vérification d'autorisation de second niveau est effectuée par le serveur. Cette vérification côté serveur doit toujours consigner les échecs d'autorisation et la propriété du client n'affecte pas cette consignation.

Le journal de tous les échecs d'autorisation est stocké sur la base de données pour permettre à ces échecs d'être audités ultérieurement. La table `AuthorisationLog` contient le nom d'utilisateur et l'identificateur de sécurité de l'autorisation ayant échoué, ainsi qu'un horodatage indiquant le moment où l'échec s'est produit. Pour plus d'informations sur la table `AuthorisationLog`, consultez la rubrique 11.15, «Analyse de la table de base de données `AuthorisationLog`», à la page 44.

Chapitre 4. Cryptographie dans Cúram

4.1 Présentation

Dans IBM Cúram Social Program Management, la cryptographie fait globalement référence aux deux types de fonctionnalité associés à la sécurisation des systèmes Cúram :

1. chiffrements - chiffrement bidirectionnel des mots de passe utilisés à différents points de traitement
2. traitements - hachage unidirectionnel (ou traitement) des mots de passe (par ex., ceux utilisés lors de la connexion)

Les valeurs de configuration du comportement cryptographique peuvent être sélectionnées par l'utilisateur via un fichier de propriétés (`CryptoConfig.properties`) pour apporter des niveaux de contrôle et de sécurité optimaux dans votre installation Cúram. Cette souplesse permet de s'ajuster aux normes de sécurité changeantes. Voir Chapitre 12, «Personnalisation de la cryptographie», à la page 45 pour plus d'informations sur la configuration et la personnalisation de la cryptographie.

Il est recommandé aux utilisateurs existants d'IBM Cúram Social Program Management de mettre à niveau les mots de passe système (nouveau chiffrement) et utilisateur (nouveau traitement) à partir des valeurs par défaut prêtes à l'emploi existantes pour améliorer votre sécurité. La modification des mots de passe utilisateur est un changement plus substantiel. Vous pouvez effectuer ces deux changements indépendamment, comme décrit dans les rubriques associées. Si vous souhaitez accepter un niveau de sécurité réduit, vous pouvez, à vos risques, choisir de ne pas modifier les mots de passe système et utilisateur. Toutefois, cela n'est pas recommandé.

Les configurations cryptographiques prises en charge sont les suivantes :

1. AES : 128, 192, 256 (conformes FIPS 140-2 et SP800-131a) ;
2. Triple DES à deux clés - DESede : 112 (conforme FIPS 140-2) ;
3. Triple DES à trois clés - DESede : 168 (FIPS 140-2 et SP800-131a) ;
4. Aucune configuration de cryptographie, qui est configurée en supprimant le fichier `CryptoConfig.properties`. Dans ce cas, Cúram rétablit ses paramètres de cryptographie prêts à l'emploi précédents.

Dans l'environnement d'exécution Cúram, le serveur d'application, la base de données et d'autres logiciels (par ex., serveur Web, LDAP , etc.) possèdent leur propre support cryptographique. Reportez-vous à la documentation du fournisseur en question.

4.2 Chiffrement

Le chiffrement fait référence au processus de chiffrement des mots de passe. Il s'agit donc d'un processus bidirectionnel représentant des valeurs déchiffrables. Il existe environ une dizaine de mots de passe chiffrés dans Cúram. Leur chiffrement permet de les sécuriser. Ils sont déchiffrés aux points d'utilisation nécessaires (par ex., connexion à votre système de bases de données).

4.3 Traitement

Le traitement fait référence au processus unidirectionnel de traitement des mots de passe qui ne nécessitent pas de déchiffrement. Il est utilisé pour stocker les mots de passe en vue d'une comparaison ultérieure (par exemple, connexions utilisateur Cúram). Il s'agit d'un processus unidirectionnel qui représente des valeurs non déchiffrables.

4.4 Propriétés de cryptographie

Le fichier `CryptoConfig.properties` contient des paramètres de chiffrement et de traitement. Par conséquent, ce fichier et tous les fichiers auxquels il se réfère (par ex., fichier de clés et salt) sont des éléments essentiels à la sécurité de votre système et doivent disposer de contrôles d'accès appropriés (par ex. : autorisations d'accès aux fichiers) et être spécifiquement modifiés et séparés lorsqu'ils sont utilisés pour les systèmes de production. Si les détails de ces fichiers deviennent largement connus, mais ne représentent pas nécessairement un risque de sécurité, un niveau de protection peut devenir obsolète et nécessiter une interruption pour changement de cryptographie (voir 12.2, «Personnalisation du chiffrement», à la page 45 et 12.5, «Personnalisation du traitement», à la page 47).

Rubriques connexes :

- 4.5, «Paramètres de chiffrement Cúram»
- 4.6, «Paramètres de traitement Cúram», à la page 21

4.5 Paramètres de chiffrement Cúram

Différents mots de passe et différentes configurations Cúram sont stockés dans un format chiffré prêt à l'emploi (OOTB).

La configuration de cryptographie Cúram est prête à l'emploi. Il est recommandé de modifier ces paramètres par rapport à vos exigences de sécurité locales. Par exemple, les paramètres prêts à l'emploi peuvent être adaptés au développement. Toutefois, pour les environnements de production, il est fortement recommandé de les modifier (par ex., en changeant la clé confidentielle de chiffrement).

Les paramètres de chiffrement sont stockés dans le fichier `CryptoConfig.properties`. Les propriétés et leurs valeurs sont les suivantes :

- `curam.security.crypto.cipher.algorithm`
 - **Valeurs valides** : Dans la documentation JCE, par exemple : <http://docs.oracle.com/javase/6/docs/technotes/guides/security/StandardNames.html#Cipher>. Les chiffrements pris en charge sont la norme AES et les différentes formes de norme DES.
 - **Par défaut** : AES (conforme FIPS 140-2 et SP800-131a)
- `curam.security.crypto.superseded.cipher.algorithm`
 - **Valeurs valides** : Voir `curam.security.crypto.cipher.algorithm`
 - **Valeur par défaut** : Aucune
 - **Objectif** : apporter de la souplesse pour prendre en charge une période de mise à niveau/migration avec code personnalisé (par ex. : programme de traitement par lots) via l'API `curam.util.security.EncryptionUtil.decryptSupersededPassword()`.
- `curam.security.crypto.cipher.keystore.location`
 - **Valeurs valides** : chemin vers un fichier de clés contenant la clé confidentielle. Il peut s'agir d'une spécification de chemin absolue ou relative au chemin de classes (par ex. : `CuramSample.keystore`).
 - **Valeur par défaut** : Aucune
- `curam.security.crypto.cipher.keystore.storepass`
 - **Valeurs valides** : Selon la commande **keytool** JDK.
 - **Valeur par défaut** : `password`
 - **Objectif** : Spécifier le mot de passe utilisé pour accéder au fichier de clés.
- `curam.security.crypto.cipher.provider.class`
 - **Valeurs valides** : Nom qualifié complet d'une classe de fournisseurs de cryptographie JCE.
 - **Valeur par défaut** : Aucune
 - **Objectif** : Option permettant d'utiliser un autre fournisseur conforme aux normes.

Cette fonctionnalité de chiffrement s'applique aux propriétés décrites dans 4.7, «Mots de passe chiffrés», à la page 22.

Ces paramètres cryptographiques Cúram sont, par défaut, prêts à l'emploi et représentent les changements que les installations Cúram existantes doivent effectuer dans le *Guide de mise à niveau Cúram*.

4.6 Paramètres de traitement Cúram

Lorsqu'ils ne sont pas appelés avec l'authentification d'identité seule, les utilisateurs Cúram internes et externes sont authentifiés à l'aide d'une connexion par formulaire et le mot de passe saisi dans le formulaire est chiffré, puis comparé à la valeur de traitement stockée dans la base de données pour l'utilisateur.

Remarque : Ce traitement ne s'applique pas aux utilisateurs authentifiés dans des systèmes tiers tels que LDAP.

La configuration de cryptographie Cúram est prête à l'emploi. Il est recommandé de modifier ces paramètres par rapport à vos exigences de sécurité locales. Par exemple, les paramètres prêts à l'emploi peuvent être adaptés au développement. Toutefois, pour les environnements de production, il est fortement recommandé de les modifier (par ex., valeur du sel de traitement chiffrée).

Les paramètres de chiffrement sont stockés dans le fichier `CryptoConfig.properties`. Les propriétés et leurs valeurs sont les suivantes :

- `curam.security.crypto.digest.algorithm`
 - **Valeurs valides :** Dans la documentation JCE, par exemple : <http://docs.oracle.com/javase/6/docs/technotes/guides/security/StandardNames.html#MessageDigest>. Les traitements pris en charge sont les variantes SHA (1, 256, etc.) et MD5.
 - **Par défaut :** SHA-256 (conforme FIPS 140-2 et SP800-131a)
 - **Objectif :** Spécification de l'algorithme de traitement.
- `curam.security.crypto.digest.salt.location`
 - **Valeurs valides :** Chemin identifiant le fichier contenant le sel de traitement confidentiel chiffré.
 - **Valeur par défaut :** Aucune
 - **Objectif :** Fichier facultatif pour spécifier le sel (chiffré) en vue du traitement.
- `curam.security.crypto.digest.iterations`
 - **Valeurs valides :** 0 ou entier positif.
 - **Valeur par défaut :** 0
 - **Objectif :** Les valeurs supérieures apportent généralement une meilleure sécurité, mais au prix du traitement (par ex.,).

Un ensemble de propriétés obsolètes correspondantes apporte de la souplesse lors de la migration d'un ensemble de paramètres ou de normes de traitement vers un autre. Les propriétés suivantes ont une fonction similaire à celle de leurs homologues ci-dessus, mais sont utilisées par la fonctionnalité de chiffrement Cúram pour prendre en charge anciens et nouveaux paramètres pendant la période de migration :

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

L'utilisation et le comportement des propriétés obsolètes sont contrôlés par la propriété `curam.security.convertsupersededpassworddigests.enabled`, elle-même gérée par l'interface utilisateur Administration des propriétés. Voir 12.7, «Utilisation des paramètres de traitement obsolètes pour une période de migration», à la page 48 pour plus d'informations sur les propriétés obsolètes.

4.7 Mots de passe chiffrés

Les mots de passe suivants sont chiffrés dans Cúram :

- `Bootstrap.properties` :
 - `curam.db.password` - mot de passe de base de données
 - `curam.searchserver.sync.password` - voir *Cúram Generic Search Server* pour plus d'informations
- `AppServer.properties` : (ce fichier de propriétés est généralement utilisé pour configurer les serveurs de test et n'est pas adapté aux systèmes de production)
 - `security.password` - mot de passe de console d'administration de serveur d'applications
 - `curam.security.credentials.async.password` - remplace `runas.password` property
- `Application.prx` - Les descriptions de propriétés individuelles apparaissent telles que documentées avec les propriétés dans l'interface utilisateur Curam Property Administration :
 - `curam.security.credentials.dbtojms.password` - (en association avec `curam.security.credentials.dbtojms.username`), qui remplace les API d'interface `curam.omega3.DBtoJMSCredentialsIntf` précédemment utilisés pour fournir des données d'identification personnalisées pour DB-TO-JMS
 - `curam.security.credentials.ws.password` (en association avec `curam.security.credentials.ws.username`), qui remplace les paramètres de données d'identification par défaut des services Web par défaut de temps de génération.
 - `curam.meeting.request.reply.password` - (mot de passe SMTP)
 - `curam.ldap.password`
 - `curam.citizenworkspace.password.protection.key`
- `BIBootstrap.properties` - utilisateurs BIRT uniquement ; voir le manuel *Cúram Business Intelligence BIRT - Guide de développement*:
 - `curamsource.db.password`
 - `central.db.password`
 - `centraldm.db.password`
- Services Web - Voir :
 - `ws_inbound.xml` - `<ws_service_password>`
 - `services.xml` - `<parameter name="jndiPassword">`
- CTM - Cúram Transport Manager :
- La colonne Mot de passe de la table `TargetSystemService` contient un mot de passe chiffré

Chapitre 5. Mise en cache des données de sécurité

5.1 Présentation

Ce chapitre décrit le cache de sécurité Cúram, qui stocke toutes les données d'autorisation pour un utilisateur. Des informations détaillées sur le cache WebSphere et la manière dont il affecte l'authentification d'un utilisateur lors de la connexion sont également incluses dans ce chapitre.

5.2 Cache de sécurité Cúram

Les informations de sécurité issues des tables de base de données prenant en charge les profils indiqués dans la rubrique 3.2, «Utilisateurs, rôles et groupes», à la page 15 sont mises en cache par l'infrastructure. Cela permet d'optimiser la recherche et la récupération des données au cours du processus d'autorisation.

Pour optimiser les performances, le cache est chargé à la demande au fur et à mesure que les demandes d'autorisation de sécurité arrivent dans l'application et constitue une ressource partagée. Pour le code d'application, le cache est une ressource protégée et n'est pas accessible directement. Il est accessible pour les requêtes uniquement, via l'interface d'autorisation (`curam.util.security.Authorisation`) qui permet à un développeur d'implémenter une procédure d'autorisation personnalisée. Pour plus d'informations, consultez la rubrique 11.14, «Autorisation de nouveaux types d'identificateurs de sécurité», à la page 44.

Lorsque la propriété `curam.security.casesensitive` est définie sur Faux, le cache de sécurité enregistre tous les noms d'utilisateur en majuscules et toutes les requêtes sur le cache remplacent automatiquement le nom d'utilisateur spécifié par son équivalent en majuscules. Il convient également de noter que l'existence de noms d'utilisateurs insensibles à la casse en doublons entraîne une erreur fatale lors de l'initialisation du cache de sécurité. Pour plus d'informations, consultez la rubrique 10.4, «Changement de la sensibilité à la casse du nom d'utilisateur», à la page 37.

5.3 Actualisation du cache

Les données de sécurité étant vitales au fonctionnement d' IBM Cúram Social Program Management, la mémoire cache doit être actualisée dès que des changements sont apportés aux tables de base de données relatives à la sécurité. L'actualisation du cache de sécurité Cúram est un processus asynchrone.

5.4 Echec d'actualisation du cache

L'actualisation du cache de sécurité Cúram est déclenchée par un redémarrage de l'application ou par l'administrateur système (`sysadmin`) via les écrans Cúram Administration, par conséquent, l'administrateur ne reçoit aucun commentaire si le rechargement du cache échoue. L'obligation de vérifier les journaux système ou de vérifier manuellement l'application après une actualisation pour vérifier sa réussite peut s'avérer fastidieuse. Il est donc recommandé d'implémenter l'interface de rappel facultative pour fournir des commentaires en cas d'échec de rechargement du cache. Pour plus d'informations, consultez la rubrique 10.8, «Ajout de l'interface de rappel d'échec de l'actualisation du cache», à la page 38.

5.5 Comportement de la mise en cache WebSphere

WebSphere met en mémoire cache les informations utilisateur et les données d'identification dans son propre cache de sécurité. Le module de connexion Cúram n'est pas appelé tant qu'une entrée utilisateur est valide dans ce cache. La durée d'invalidation par défaut de ce cache de sécurité est de dix minutes, où l'utilisateur a été inactif pendant dix minutes.

Par exemple, la première fois qu'un utilisateur se connecte à l'application à partir du client Web, il doit entrer son nom d'utilisateur et son mot de passe. Le module de connexion Cúram est appelé et authentifie les informations indiquées. Si le même utilisateur ouvre un deuxième nouveau navigateur Web et tente d'accéder à l'application, il doit à nouveau entrer son nom d'utilisateur et son mot de passe. Lorsque WebSphere reçoit ces informations, il interroge le cache de sécurité pour déterminer si le nom d'utilisateur et le mot de passe existent déjà dans le cache. S'ils existent et que le mot de passe correspond, WebSphere n'interroge pas les modules de connexion.

L'impact de ce comportement est que les modifications apportées aux restrictions ou au mot de passe ne sont pas applicables tant que l'utilisateur n'est pas invalidé à partir du cache de sécurité WebSphere.

Pour plus d'informations, voir le *Centre de documentation du serveur d'applications WebSphere*.

Chapitre 6. Sécurité des autres clients

6.1 Présentation

Il existe des processus qui peuvent ne pas être associés à un utilisateur connecté spécifique. Ceux-ci comprennent d'autres clients, par exemple les processus non Web comme le traitement par lots, les services Web et le traitement différé. Étant donné que tout processus qui interagit avec une application IBM Cúram Social Program Management doit être authentifié, un utilisateur valide doit exister pour chaque processus. Les sections suivantes fournissent des informations détaillées sur les utilisateurs qui doivent exister sur la table des utilisateurs Cúram ainsi que des détails sur les processus qui dépendent de ces utilisateurs.

6.2 Utilisateurs Cúram obligatoires

Un certain nombre d'utilisateurs doit toujours exister dans la table de base de données des utilisateurs Cúram. Ces utilisateurs sont nécessaires aux processus applicatifs tels que le traitement différé et le flux de travaux. Si ces utilisateurs n'existent pas, alors l'authentification échoue et ces processus échouent ensuite.

Les noms d'utilisateur et les mots de passe pour chacun des processus ci-dessous sont les données d'identification prêtes à l'emploi par défaut et il est recommandé de les modifier pour des raisons de sécurité.

Ces utilisateurs sont les suivants :

- SYSTEM

L'utilisateur SYSTEM est l'utilisateur sous lequel les messages JMS sont exécutés. Cet utilisateur doit exister et le nom d'utilisateur est sensible à la casse. Pour plus d'informations, reportez-vous à la rubrique 6.5, «Messagerie JMS», à la page 26.

- DBTOJMS

L'utilisateur DBTOJMS est l'utilisateur par défaut sous lequel le déclencheur du transfert de la base de données vers JMS (DBToJMS) pour le traitement par lots est exécuté. Cet utilisateur doit exister et le nom d'utilisateur est sensible à la casse. Pour plus d'informations, reportez-vous à la rubrique 6.4, «Traitement par lots», à la page 26.

- WEBSVCS

L'utilisateur WEBSVCS est l'utilisateur par défaut sous lequel les services Web sont exécutés. Cet utilisateur doit exister et le nom d'utilisateur est sensible à la casse. Pour plus d'informations, reportez-vous à la rubrique 6.3, «Services Web».

6.3 Services Web

Pour Apache Axis2 (l'implémentation recommandée pour les services Web), il existe des données d'identification par défaut pour l'authentification. Un utilisateur peut modifier ces données d'identification à un niveau global ou pour un service si nécessaire. Afin de garantir que les services Web ne sont pas vulnérables face à une violation de sécurité, cet utilisateur par défaut n'est pas autorisé à accéder aux services Web par défaut. Pour l'autorisation, un service Web doit être associé à un groupe de sécurité ou à un rôle de sécurité lié à l'utilisateur (par exemple WEBSVCS) pour y accéder. La vérification de l'autorisation de l'utilisateur est un processus manuel. Pour plus d'informations sur les services Web, consultez la section *Personnalisation de la fonctionnalité d'exécution du récepteur* du *Guide de services Web Cúram* ainsi que le chapitre sur l'autorisation dans ce manuel.

Pour Apache Axis 1.4, c'est-à-dire les services Web existants, une fois qu'un processus est modélisé en tant que service Web, ce service Web est automatiquement connecté à l'application à l'aide des données d'identification par défaut. Cet utilisateur par défaut est configuré pour l'autorisation automatiquement, c'est-à-dire que l'utilisateur a accès au service Web créé. La prudence est donc conseillée lorsqu'une classe devient visible en tant que service Web. Reportez-vous à la section *Services Web entrants existants* du *Guide des services Web Cúram*.

Il existe un certain nombre d'autres rubriques associées à la sécurité des services Web (par exemple, le chiffrement des données) qui utilisent Rampart. Pour plus d'informations, consultez le *Guide des services Web Cúram*.

6.4 Traitement par lots

Le programme de lancement par lots ne nécessitant pas que le serveur d'applications soit en cours d'exécution, il n'effectue aucune authentification ni autorisation de niveau d'application. Il doit uniquement s'authentifier par rapport à la base de données. Les mêmes données d'identification que celles utilisées par le serveur d'applications (se trouvant dans `%SERVER_DIR%/project/properties/Bootstrap.properties`) sont utilisées par le programme de lancement par lots pour se connecter à la base de données et exécuter des programmes batch.

Le programme de lancement par lots ou les programmes batch peuvent éventuellement déclencher le serveur d'applications afin de commencer un transfert de la base de données vers JMS. Cela implique la connexion et l'appel d'une méthode sur le serveur, qui à son tour nécessite un nom d'utilisateur et un mot de passe valides. L'opération de transfert de la base de données vers JMS utilise les données d'identification par défaut ; par conséquent, le compte DBTOJMS doit exister sur la table des utilisateurs Cúram et doit être activé et se voir affecter le rôle 'SYSTEMROLE' pour permettre l'autorisation. Le transfert de la base de données vers JMS de l'environnement local correspond à l'environnement local par défaut pour cet utilisateur, comme indiqué dans la zone 'defaultLocale' sur la table des utilisateurs.

Pour plus d'informations sur le changement d'utilisateur pour le transfert de la base de données vers JMS, consultez la section *Remarques sur la sécurité* du *Guide de traitement par lots Cúram*.

La propriété `batch.username` peut être utilisée pour indiquer le nom d'utilisateur pour les opérations exécutées par le programme de lancement par lots. Celle-ci est définie à l'aide du paramètre `-D`. Par exemple : `build runbatch -Dbatch.username=admin`

6.5 Messagerie JMS

Les messages JMS sont utilisés à des fins de communication par les traitements différés et le flux de travaux. Etant donné que les messages JMS sont déclenchés par le serveur d'applications et qu'ils ont besoin d'interagir avec l'application IBM Cúram Social Program Management, des données d'identification Cúram valides doivent exister. Le compte utilisateur SYSTEM doit exister sur la table des utilisateurs Cúram et doit être activé et se voir affecter le rôle 'SYSTEMROLE' pour assurer l'autorisation. L'environnement local pour les messages JMS correspond à l'environnement local par défaut pour cet utilisateur, comme indiqué dans la zone 'defaultLocale' sur la table des utilisateurs.

Il est possible de modifier le nom d'utilisateur SYSTEM pendant ou après le déploiement de l'application. Pour plus d'informations, consultez le *Guide de déploiement de serveur Cúram* correspondant au serveur d'applications approprié.

6.6 Traitement différé

Un traitement différé dans IBM Cúram Social Program Management correspond à une méthode métier qui est appelée de façon asynchrone. Comme les traitements différés interagissent avec l'application, des données d'identification Cúram valides doivent exister. Le compte utilisateur SYSTEM doit exister sur la table des utilisateurs Cúram et doit être activé et se voir affecter le rôle 'SYSTEMROLE' pour assurer l'autorisation. L'environnement local pour les traitements différés correspond à l'environnement local par défaut pour cet utilisateur, comme indiqué dans la zone 'defaultLocale' sur la table des utilisateurs. En cas de test d'unité hors ligne des traitements différés, le nom d'utilisateur est vide et l'environnement local effectif est l'environnement local par défaut du serveur IBM Cúram Social Program Management.

Chapitre 7. Applications d'utilisateurs externes

7.1 Présentation

L'application IBM Cúram Social Program Management par défaut et prête à l'emploi est activée pour les utilisateurs internes. Les utilisateurs internes sont des utilisateurs qui existent sur la table de base de données des utilisateurs Cúram. Un utilisateur interne typique est par exemple un responsable du dossier qui crée et gère les demandes des participants et dispose de l'accès complet à l'application.

L'infrastructure offre des fonctionnalités permettant d'authentifier et d'autoriser ces utilisateurs internes.

Il existe également le concept d'un utilisateur qui doit accéder en toute sécurité à certaines parties de l'application IBM Cúram Social Program Management. Cet utilisateur se trouve en dehors de l'organisation. Son accès est limité. Les utilisateurs de ce type sont considérés comme des utilisateurs externes et l'authentification de ces derniers est entièrement personnalisable à l'aide du point d'ancrage de sécurité d'accès externe fourni. Les utilisateurs externes étant traités différemment des utilisateurs internes, une application Web spécifique est obligatoire pour les utilisateurs externes.

7.2 Applications d'utilisateurs externes

Lors du développement d'une application pour un utilisateur externe, les éléments suivants doivent être implémentés:

- Application client d'utilisateur externe, par ex. fichier EAR contenant l'application client Web.
- Fichier `logon.jsp` personnalisé, où l'application externe doit transmettre un paramètre `user_type` indiquant qu'un utilisateur externe se connecte.
- Une classe personnalisée étendant `curam.util.security.PublicAccessUser` et nécessitant l'implémentation de l'interface `curam.util.security.ExternalAccessSecurity` doit être fournie. Cette classe abstraite comprend les méthodes responsables de l'authentification et de l'autorisation d'un utilisateur externe.

Il existe des types d'utilisateurs internes et externes. Il peut également y avoir différents types d'utilisateurs externes. Par exemple, il peut exister un utilisateur externe de type 'PUBLIC' qui dispose d'un accès limité à une application externe. Il peut exister un autre utilisateur externe de type 'PROVIDER' qui correspond à un utilisateur externe enregistré. La possibilité de disposer de différents types d'utilisateurs externes apporte plus de flexibilité dans une application externe, en permettant un contrôle accru de l'authentification de l'utilisateur externe en fonction de son type.

7.3 Portée d'utilisateur

Il existe deux différents types, ou portées, d'utilisateur dans l'application IBM Cúram Social Program Management: utilisateurs internes et externes. Le type d'utilisateur est déterminé de l'une des manières suivantes :

- Par le cache de sécurité Cúram
Si l'utilisateur existe dans le cache de sécurité Cúram, le type est censé être interne. Si l'utilisateur n'existe pas dans le cache, le type est censé être externe. Dans ce cas (il s'agit du comportement par défaut), tous les utilisateurs, internes et externes, doivent être uniques.
- Par l'interface personnalisée `UserScope`
Si l'interface personnalisée `UserScope` est implémentée. Cette interface personnalisée prévaut sur la vérification d'un utilisateur dans le cache de sécurité Cúram pour déterminer le type d'utilisateur. Pour plus d'informations, consultez la rubrique 13.15, «Détermination d'un utilisateur interne ou externe à l'aide de l'interface `UserScope`», à la page 58.

Lorsque le type d'un utilisateur est externe, l'implémentation de la méthode `curam.util.security.ExternalAccessSecurity.getSecurityRole()` permet de déterminer le rôle de l'utilisateur au lieu des rôles de sécurité internes. Pour plus d'informations sur cette méthode, consultez la rubrique 13.8, «Autorisation d'un utilisateur externe», à la page 55.

L'interface personnalisée `UserScope` est disponible pour prendre en charge les autres méthodes de détermination d'un utilisateur interne ou externe. Pour plus d'informations, consultez la rubrique 13.15, «Détermination d'un utilisateur interne ou externe à l'aide de l'interface `UserScope`», à la page 58.

7.4 Déploiement d'une application externe

Lors du déploiement d'une application sur un serveur d'application, la configuration de la sécurité du serveur d'applications s'applique à toutes les applications IBM Cúram Social Program Management déployées sur cette instance de serveur d'applications. Par conséquent, la prudence est impérative lorsque l'architecture de déploiement est envisagée pour plusieurs applications. Cet aspect est important pour décider si une application interne ou externe doit être déployée sur la même instance de serveur d'applications.

Les aspects à étudier sont par exemple :

- L'identité seule est-elle utilisée pour les utilisateurs internes ?
- Un autre mécanisme d'authentification est-il utilisé, par exemple le protocole LDAP ?
- Les utilisateurs internes et externes sont-ils tous authentifiés par le protocole LDAP ?

Les réponses aux questions ci-dessus déterminent la définition des propriétés du serveur d'applications (autrement dit, les propriétés indiquées dans le fichier `AppServer.properties`), qui elles-mêmes affectent le comportement du module de connexion JAAS Cúram. Ces aspects orientent également l'implémentation de la classe `curam.util.security.PublicAccessUser` et de l'interface `curam.util.security.ExternalAccessSecurity` pour les utilisateurs externes.

Les propriétés du serveur d'applications dans le module de connexion JAAS Cúram permettent un contrôle plus précis de l'authentification des types d'utilisateurs. Les utilisateurs externes et internes peuvent être authentifiés différemment, comme c'est le cas des différents types d'utilisateurs externes, dans une situation où les applications internes et externes sont déployées sur le même serveur d'applications. Ces propriétés comprennent les éléments suivants :

- `curam.security.user.registry.disabled.types`
Définissez cette propriété dans une liste séparée par des virgules répertoriant les types d'utilisateur pour lesquels le registre d'utilisateurs *n'est pas* interrogé (par ex., l'implémentation dans la méthode `PublicAccessUser.authenticate()` est responsable de l'authentification de l'utilisateur externe de ce type. Par exemple, le protocole LDAP peut être configuré en tant que registre d'utilisateurs.
- `curam.security.user.registry.enabled.types`
Définissez cette propriété dans une liste séparée par des virgules répertoriant les types d'utilisateur pour lesquels le registre d'utilisateur *est interrogé* (par ex., l'implémentation dans la méthode `PublicAccessUser.authenticate()` ne doit pas authentifier complètement l'utilisateur. Le registre d'utilisateurs est responsable de l'authentification de ce type d'utilisateur externe. Par exemple, le protocole LDAP peut être configuré en tant que registre d'utilisateurs, et dans ce cas, le protocole LDAP peut être responsable de l'authentification de ces types d'utilisateurs externes.

Ces propriétés dépendent de l'implémentation de la classe `curam.util.security.PublicAccessUser` et de l'interface `ExternalAccessSecurity`.

Tenez compte des exemples d'exigences du projet ci-après :

- Un utilisateur interne doit s'authentifier auprès du protocole LDAP.
- Un utilisateur externe de type 'EXT_PUBLIC' doit s'authentifier auprès d'IBM Cúram Social Program Management et non du protocole LDAP.
- Un utilisateur externe de type 'EXTERNAL' doit s'authentifier uniquement auprès du protocole LDAP et non auprès d'IBM Cúram Social Program Management.
- Les applications internes et externes sont déployées sur la même instance de serveur d'applications.

Les paramètres suivants peuvent prendre en charge l'exemple ci-dessus :

- `curam.security.check.identity.only` défini sur vrai
- `curam.security.user.registry.disabled.types=EXT_PUBLIC`.

De même que les propriétés définies, l'extension `PublicAccessUser` extension (et l'implémentation de `curam.util.security.ExternalAccessSecurity`) doit posséder la logique permettant de traiter les différents types d'utilisateur et leur mode d'authentification.

Chapitre 8. Utilisation d'une connexion unique

8.1 Présentation

Le nombre d'applications dans une entreprise entraîne souvent une augmentation du nombre de noms d'utilisateur et de mots de passe en cours d'utilisation, ce qui génère une mauvaise expérience utilisateur et un coût supplémentaire lié à leur maintenance. Plusieurs noms d'utilisateur et mots de passe menacent également la sécurité, étant donné que les utilisateurs choisissent des mots de passe très simples ou écrivent leurs mots de passe dans des emplacements faciles à trouver. Pour les administrateurs système, les applications supplémentaires impliquent un effort accru de maintenance du répertoire et la prise en charge d'appels au centre d'assistance plus nombreux pour réinitialiser les mots de passe. Certains problèmes liés aux applications supplémentaires peuvent être résolus à l'aide d'une fonctionnalité de connexion unique. La fonctionnalité de connexion unique permet aux utilisateurs d'accéder à plusieurs applications sécurisées en s'authentifiant une seule fois.

Remarque : Le terme sécurisé s'applique aux applications qui nécessitent l'authentification des utilisateurs avant qu'ils accèdent à la fonctionnalité.

La connexion unique est prise en charge pour les serveurs d'applications pris en charge, en permettant l'utilisation d'autres mécanismes simultanément au module de connexion Cúram. L'implémentation d'une solution de connexion unique dépend de l'implémentation personnalisée. Il est recommandé d'utiliser un outil tiers, par exemple IBM Tivoli™ ou CA SiteMinder.

Ce chapitre décrit les propriétés du serveur d'applications permettant l'utilisation d'une solution de connexion unique.

8.2 Connexion unique avec WebSphere

Lorsque la connexion unique est requise avec WebSphere, elle peut être obtenue à l'aide du protocole LTPA de WebSphere et des modules de connexion personnalisés supplémentaires. Le protocole d'authentification LTPA entraîne la création d'un jeton pour un utilisateur authentifié. Dans WebSphere, un jeton est généré une fois que les données d'identification sont ajoutées pour un utilisateur authentifié. Ce jeton est ensuite utilisé pour récupérer les informations d'identité relatives à un utilisateur authentifié dans un environnement de connexion unique.

La sécurité est implémentée en tant que module de connexion Cúram au sein d'une chaîne de modules de connexion configurés dans WebSphere. Au moins un de ces modules de connexion doit ajouter des données d'identification pour l'utilisateur. Par défaut, le module de connexion Cúram ajoute des données d'identification pour un utilisateur authentifié. Par conséquent, le registre d'utilisateurs WebSphere configuré géré par un module de connexion ultérieur n'ajoute pas de données d'identification. L'approche recommandée pour implémenter une solution de connexion unique consiste à ajouter un module de connexion personnalisé sur la chaîne de modules de connexion.

Il est possible de désactiver l'ajout de données d'identification pour un utilisateur non authentifié, ce qui permet l'implémentation d'une solution de connexion unique.

Le module de connexion JAAS Cúram pour WebSphere vérifie s'il existe un jeton LTPA dans WebSphere à l'aide du rappel WSCredTokenCallbackImpl pour WebSphere. Si ce jeton existe et est valide, aucune authentification n'est effectuée par le module de connexion Cúram.

Des données d'identification peut être ajoutées a registre d'utilisateurs WebSphere. Les données d'identification incluent des informations d'authentification sur l'utilisateur qui se connecte, notamment l'identificateur unique de l'utilisateur. WebSphere vérifie qu'il existe des données d'identification pour un utilisateur après que tous les modules de connexion du système configuré ont été exécutés. S'il existe des données d'identification, le registre d'utilisateurs WebSphere n'est pas interrogé. Les données d'identification ne sont pas ajoutées par le module de connexion JAAS Cúram si les paramètres suivants sont présents :

- la propriété `curam.security.check.identity.only` est définie sur Vrai.
- la propriété `curam.security.user.registry.enabled` est définie sur Vrai.

Comme indiqué dans 7.4, «Déploiement d'une application externe», à la page 30, certaines propriétés relatives au type d'utilisateur externe contrôlent si des données d'identification sont ajoutées à WebSphere pour un type d'utilisateur externe spécifique. Celles-ci comprennent :

- propriété `curam.security.user.registry.enabled.types`.
- propriété `curam.security.user.registry.disabled.types`.

Ces propriétés offrent un contrôle plus précis de l'authentification pour les types d'utilisateurs externes.

Si le module de connexion JAAS Cúram n'ajoute pas de données d'identification, le registre d'utilisateur WebSphere est invité à tenter d'ajouter des données d'identification pour l'utilisateur.

8.3 Connexion unique avec WebLogic Server

Lorsque la connexion unique est requise avec WebLogic Server, elle peut être obtenue grâce au fournisseur d'authentification WebLogic Server ou à un fournisseur d'authentification personnalisé. Pour plus d'informations sur les fournisseurs d'authentification, consultez la documentation WebLogic Server. WebLogic Server nécessite que les données d'identification/données principales et le groupe auquel l'utilisateur appartient soient ajoutés par le fournisseur d'authentification configuré. Pour une solution de connexion unique, le module de connexion JAAS Cúram n'ajoute pas de données d'identification à l'objet JAAS pour permettre à un autre fournisseur d'authentification d'ajouter des données d'identification.

Les données d'identification ne sont pas ajoutées si les paramètres suivants sont présents :

- `curam.security.check.identity.only` est défini sur Vrai.
- `curam.security.user.registry.enabled` est défini sur Vrai.

Comme indiqué dans 7.4, «Déploiement d'une application externe», à la page 30, certaines propriétés relatives au type d'utilisateur externe contrôlent si des données d'identification sont ajoutées à WebLogic Server pour un type d'utilisateur spécifique. Celles-ci comprennent :

- propriété `curam.security.user.registry.enabled.types`.
- propriété `curam.security.user.registry.disabled.types`.

Ces propriétés offrent un contrôle plus précis de l'authentification pour les types d'utilisateurs externes.

La responsabilité d'ajouter des données d'identification est laissée à un autre fournisseur d'authentification, c'est-à-dire le fournisseur d'authentification principal, afin d'authentifier l'utilisateur. Dans un scénario de connexion unique, un seul fournisseur parmi les fournisseurs d'authentification doit ajouter des données d'identification à l'objet JAAS dans le cadre de la méthode `commit()` du module de connexion pour un utilisateur.

Chapitre 9. Autres remarques relatives à la sécurité

9.1 Présentation

Un autre enjeu de sécurité important réside dans la protection du contenu. En effet, celui-ci est entré, affiché et transféré sur le réseau pour l'application IBM Cúram Social Program Management. La configuration par défaut utilise la couche SSL fournie par le serveur d'applications pour sécuriser le contenu lors de son transfert.

En outre, au cours du cycle de vie de développement, des produits de pointe sont utilisés pour surveiller régulièrement les vulnérabilités de sécurité dans l'application. Ces vulnérabilités potentielles incluent par exemple : scriptage de site croisé et injection SQL. Ces menaces sont gérées dans l'infrastructure lorsqu'elles sont découvertes.

Il est recommandé que les clients assurent une surveillance de sécurité similaire de leur application.

9.2 Paramètres SSL pour l'application

Les paramètres SSL sont définis sur la valeur par défaut pour l'accès à l'application Web. Cela assure une connexion SSL sécurisée entre le client et le serveur et garantit également le chiffrement des données. Les paramètres SSL sont activés pour le client via les paramètres du fichier `web.xml` pour l'application client Web. Les paramètres SSL sont activés au niveau du serveur d'applications par les paramètres de WebLogic Server et WebSphere . Ces paramètres des serveurs d'applications sont définis via les scripts de configuration IBM Cúram Social Program Management.

Important : Les scripts de configuration garantissent que les paramètres SSL sont activés par défaut, toutefois, il s'agit d'une configuration par défaut qui doit être mise à jour et de nouveaux certificats doivent être établis pour le protocole SSL.

Il est recommandé de laisser les paramètres SSL activés pour accéder à l'application IBM Cúram Social Program Management ; cependant, en fonction des configurations de projet spécifiques, il peut être nécessaire de désactiver les paramètres SSL de l'application.

Il est possible de désactiver les paramètres SSL mais cela n'est pas recommandé. Pour plus d'informations, consultez la rubrique 10.9, «Désactivation des paramètres SSL de l'application», à la page 38.

Chapitre 10. Personnalisation de l'authentification

10.1 Personnalisation de la page de connexion

L'écran de connexion prêt à l'emploi par défaut est représenté par le fichier `logon.jsp` situé dans le répertoire `lib/curam/web/jsp` de l'environnement Client Development Environment for Java (CDEJ). Le fichier `logon.jsp` peut être personnalisé en créant une copie du fichier prêt à l'emploi et en plaçant ce dernier dans un dossier `webclient/components/<custom>/WebContent`, où `<custom>` représente le nom du composant du client Web personnalisé.

La section sur les pages de connexion du *Manuel de référence du client Web Cúram* contient des instructions concernant les éléments à conserver dans le fichier `logon.jsp`. Consultez-la pour plus d'informations.

10.2 Application d'un style à la page de connexion

Des changements de style peuvent être appliqués au fichier `logon.jsp` de la manière habituelle, c'est-à-dire en ajoutant la feuille de style en cascade appropriée à un fichier `.css` dans le composant personnalisé. Pour plus d'informations sur le style, consultez le *Manuel de référence du client Web Cúram*.

10.3 Activation des noms d'utilisateurs avec caractères étendus pour WebLogic Server

Si le serveur d'applications WebLogic Server n'est pas utilisé, cette section peut être ignorée.

Si vous possédez des noms d'utilisateur ou des mots de passe comportant des caractères étendus (par ex. "üßer"), WebLogic Server fournit un attribut de propriété `j_character_encoding`, qui doit être ajouté à la page de connexion par formulaire `logon.jsp`. Pour plus d'informations, consultez la documentation WebLogic Server. L'attribut doit être ajouté à l'élément de table dans le fichier `logon.jsp`, comme illustré ci-dessous : 10.3, «Activation des noms d'utilisateurs avec caractères étendus pour WebLogic Server».

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

Figure 6. Prise en charge des connexions avec caractères étendus par WebLogic Server

10.4 Changement de la sensibilité à la casse du nom d'utilisateur

La propriété `curam.security.casesensitive` contrôle la sensibilité à la casse des noms d'utilisateurs. Par défaut, celle-ci est définie sur `vrai` dans le fichier `Application.prx`. Lorsqu'elle est définie sur `faux` dans le fichier `Application.prx`, les mécanismes d'authentification et d'autorisation ignorent la casse du nom d'utilisateur.

Pour plus d'informations sur le fichier `Application.prx`, consultez le chapitre *Paramètres de configuration Cúram* du *Guide de développement de serveur Cúram*.

10.5 Ajout de vérifications personnalisées au processus d'authentification

Pour ajouter des vérifications personnalisées, l'interface `curam.util.security.CustomAuthenticator` doit être implémentée. Cette interface contient une méthode - `authenticateUser()`. La méthode `authenticateUser()` est appelée à la fois pour l'authentification par défaut et pour l'authentification d'identité uniquement. Les résultats de cette méthode sont censés constituer une entrée de la table de codes `curam.util.codetable.SECURITYSTATUS`. En cas d'authentification réussie, le résultat doit être `curam.util.codetable.SECURITYSTATUS.LOGIN`.

Pour les échecs d'authentification, n'importe quel résultat, y compris NULL, peut être retourné. Il est recommandé cependant d'utiliser un autre code de la table de codes `curam.util.codetable.SECURITYSTATUS`. Cette table de codes peut être étendue pour inclure des codes personnalisés comme indiqué dans le chapitre sur les tables de codes du *Guide de développement de serveur Cúram*.

Après que les vérifications personnalisées ont été appelées, le processus d'authentification met à jour les zones appropriées sur la table de base de données des utilisateurs. Par exemple, si le résultat des vérifications personnalisées n'est pas `SECURITYSTATUS.LOGIN`, le nombre d'échecs de connexion est augmenté de 1, et si le seuil d'intrusions est atteint, le compte est désactivé. Sinon, si le résultat est `SECURITYSTATUS.LOGIN`, les échecs de connexion sont redéfinis sur 0 et la zone de la dernière connexion réussie est mise à jour.

Remarque : Lorsque l'authentification d'identité uniquement est activée, les zones de la table de base de données des utilisateurs ne sont pas mises à jour, quel que soit le résultat de la vérification personnalisée.

10.6 Configuration de l'authentificateur personnalisé

Pour configurer l'application de manière à utiliser cette extension personnalisée, la propriété `curam.custom.authentication.implementation` du fichier `Application.prx` doit être définie sur le nom qualifié complet de la classe implémentant l'interface `CustomAuthenticator`.

Pour plus d'informations sur le fichier `Application.prx`, consultez le chapitre *Paramètres de configuration Cúram* du *Guide de développement de serveur Cúram*.

10.7 Configuration de l'authentification d'identité uniquement

Pour configurer l'authentification d'identité uniquement, la propriété `curam.security.check.identity.only` doit être définie sur `vrai` dans le fichier `AppServer.properties` avant d'exécuter la cible **configure**. Il est également possible de définir cette propriété une fois que l'application est déployée via la console du serveur d'applications. Pour plus d'informations sur la configuration du serveur d'applications, consultez le *Guide de déploiement de serveur Cúram* correspondant.

10.8 Ajout de l'interface de rappel d'échec de l'actualisation du cache

La nouvelle classe de rappel doit implémenter l'interface `curam.util.security.SecurityCacheFailureCallback` dans une classe ayant un constructeur par défaut public. L'implémentation du rappel est enregistrée en paramétrant la propriété d'application `curam.security.cache.failure.callback` sur le nom de la classe d'implémentation. Si la propriété n'est pas définie, aucune tentative n'est effectuée pour appeler un gestionnaire de rappel.

10.9 Désactivation des paramètres SSL de l'application

Les paramètres SSL sont définis sur la valeur par défaut pour l'accès à l'application IBM Cúram Social Program Management. Cela assure une connexion SSL sécurisée entre le client et le serveur et garantit également le chiffrement des données. Les paramètres SSL peuvent être activés et désactivés pour le client via les paramètres du fichier `web.xml` pour l'application client Web, ainsi qu'au niveau du serveur d'applications par les paramètres dans WebLogic Server et WebSphere. Ces paramètres des serveurs d'applications sont configurés via les scripts de configuration. Il est recommandé de laisser les paramètres SSL activés pour accéder à l'application ; cependant, en fonction des configurations de projets spécifiques, il peut être nécessaire de désactiver les paramètres SSL de l'application. Les sections suivantes indiquent la procédure à suivre pour ce faire.

10.10 Modification du fichier web.xml pour l'application client

Ce fichier peut être changé en modifiant la <garantie de transport> de CONFIDENTIAL en NONE dans le fichier web.xml. Remarque : cela ne désactive pas l'accès au client Web sur HTTPS, mais active un accès supplémentaire via HTTP. Pour plus d'informations sur la modification du fichier web.xml, consultez la section sur la *Personnalisation du descripteur d'application Web* du *Manuel de référence du client Web Cúram*. Un exemple de configuration de cette propriété est proposé ci-dessous :

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

10.11 Modification de la configuration du serveur d'applications

La modification de la configuration de WebSphere peut s'effectuer de deux manières. La première approche ci-dessous correspond à l'approche recommandée.

- Utilisez le port non sécurisé existant, configuré par défaut pour les services Web (approche recommandée). Cela s'applique à la fois aux connexions SSL et non SSL.
 1. Accédez à Environnement -> Hôtes virtuels -> hôte_client->Alias hôtes
 2. Cliquez sur Nouveau et entrez * pour le nom d'hôte et 9082 pour le numéro de port, puis cliquez sur OK
 3. Sur la page suivante, cliquez sur Enregistrer pour stocker votre nouvelle valeur dans la configuration du serveur. Le port 9082 correspond à la valeur *CuramWebServicesChain* configurée dans l'application client par défaut et ce port est désormais le port permettant d'accéder à l'application à l'aide de HTTP.

- Réutilisez le port SSL 9044 actuel :

Le port actuel peut être configuré en tant que port non sécurisé. Pour ce faire, les étapes à suivre sont décrites dans le *Guide de déploiement Cúram de WebSphere Application Server* - Section A.2.11, Configuration de serveur - Configuration d'accès de port. Procédez aux étapes 7 à 11 incluses. La seule différence pour l'étape 11 réside dans le fait que le modèle de chaîne de transport doit être défini sur 'Conteneur Web' (et non sur Conteneur Web sécurisé).

10.12 Analyse de la table de données AuthenticationLog

Toutes les tentatives de connexion (réussies ou non) sont consignées dans la table de base de données AuthenticationLog. Voici les lignes importantes dans cette table :

Tableau 1. Contenus du journal d'authentification

Zone	Signification
Heure entrée	Horodatage de l'entrée dans le journal.
Nom d'utilisateur	Nom d'utilisateur associé à la tentative de connexion.
altLogin	Booléen indiquant si le nom d'utilisateur représente un autre ID de connexion. Lorsque cette colonne est égale à '1' (true), la valeur de la colonne userName est un autre ID de connexion par rapport à 2.5, «Autres ID de connexion», à la page 5 ; sinon, la colonne userName représente le nom d'utilisateur de la table Users ou ExternalUser.
Echecs de connexion	Nombre d'échecs de connexion de cet utilisateur depuis sa dernière connexion réussie.
Dernière connexion	Date et heure de la dernière connexion réussie.
Statut de connexion	Statut de la tentative de connexion. Celui-ci peut être constitué des éléments suivants : <ul style="list-style-type: none">• LOGIN : Connexion réussie.• ACCDISABLE : Le compte a été désactivé explicitement.• ACCEPIRED : La date d'expiration du mot de passe a été atteinte.• PWDEXPIRED : Le nombre de jours accordé à l'utilisateur pour modifier son mot de passe a été dépassé.• BADUSER : L'utilisateur n'existe pas.• AUTHONLY : Ce statut est utilisé en cas d'authentification d'identité uniquement et indique que seules les vérifications d'autorisation sont effectuées.• BADPWD : Le mot de passe spécifié est incorrect.• BREAKIN : Un nombre défini de mots de passe incorrects a été atteint. Le compte est désactivé.• RESTRICTED : L'utilisateur n'est pas autorisé à accéder au système actuellement.• LOGEXPR : Le nombre de tentatives de connexion accordé à l'utilisateur pour modifier son mot de passe a été dépassé.• AMBIGUOUS : Le nom d'utilisateur spécifié est ambigu car il s'agit d'un doublon insensible à la casse d'un autre nom d'utilisateur.

L'interface de programme d'application LogAdmin peut être utilisée pour interroger la table de base de données AuthenticationLog. La documentation Java de cette classe doit être référencée pour plus d'informations.

Chapitre 11. Personnalisation d'autorisation

11.1 Présentation

Ce chapitre explique la méthode de configuration de l'autorisation pour les utilisateurs.

11.2 Création de mappage de données d'autorisation

Les données d'autorisation d'un utilisateur peuvent être configurées à l'aide du gestionnaire de données (fichiers DMX) ou via les écrans Cúram Administration. Pour plus d'informations sur la manière d'unifier la sécurité dans le cadre d'une entreprise, consultez le *Guide de configuration du système Cúram*.

Pour créer un nouveau rôle de sécurité pour un utilisateur, il est nécessaire de déterminer les identificateurs de sécurité auxquels l'utilisateur doit accéder. Ces identificateurs de sécurité doivent ensuite être organisés en groupes d'identificateurs de sécurité. Une fois identifiés, le rôle, les groupes et les identificateurs de sécurité doivent être configurés sur les tables de sécurité qu'ils représentent.

Les données de sécurité sont considérées comme essentielles à la configuration d'une application IBM Cúram Social Program Management. Aussi, les exemples ci-dessous illustrent l'ajout de données de sécurité au répertoire `data/initial` du composant.

11.3 Création d'un nouveau rôle de sécurité

Pour créer un nouveau rôle de sécurité, une nouvelle entrée doit être ajoutée à la table de base de données `SecurityRole` définissant l'attribut `nom de rôle`.

Pour ce faire, créez/ajoutez le fichier `SecurityRole.dmx` dans le répertoire `%SERVER_DIR%/components/<custom>/data/initial`, où `<custom>` correspond à tout nouveau répertoire créé sous des composants respectant une structure de répertoire de type `components/core`.

11.4 Création d'un nouveau groupe de sécurité

Pour créer un nouveau groupe de sécurité, une nouvelle entrée doit être ajoutée à la table de base de données `SecurityGroup` définissant l'attribut `nom de groupe`.

Pour ce faire, créez/ajoutez le fichier `SecurityGroup.dmx` dans le répertoire `%SERVER_DIR%/components/<custom>/data/initial`, où `<custom>` correspond à tout nouveau répertoire créé sous des composants respectant une structure de répertoire de type `components/core`.

11.5 Liaison du groupe de sécurité au rôle de sécurité

Le rôle de sécurité doit être lié au groupe de sécurité. Pour cela, créez une nouvelle entrée dans la table `SecurityRoleGroup` définissant les attributs `nom de rôle` et `nom de groupe`.

Pour ce faire, créez/ajoutez le fichier `SecurityRoleGroup.dmx` dans le répertoire `%SERVER_DIR%/components/<custom>/data/initial`, où `<custom>` correspond à tout nouveau répertoire créé sous des composants respectant une structure de répertoire de type `components/core`.

11.6 Création de l'identificateur de sécurité

Pour créer un nouvel identificateur de sécurité, une entrée doit être ajoutée à la table SecurityIdentifier définissant les attributs nom d'identificateur de sécurité et type d'identificateur de sécurité.

Pour ce faire, créez/ajoutez le fichier SecurityIdentifier.dmx dans le répertoire %SERVER_DIR%/components/<custom>/data/initial, où <custom> correspond à tout nouveau répertoire créé sous des composants respectant une structure de répertoire de type components/core.

11.7 Liaison du groupe de sécurité à l'identificateur de sécurité

Pour lier le groupe de sécurité à l'identificateur de sécurité, une entrée doit être ajoutée à la table SecurityGroupSID définissant les attributs nom de groupe et nom d'identificateur de sécurité.

Pour ce faire, créez/ajoutez le fichier SecurityGroupSID.dmx dans le répertoire %SERVER_DIR%/components/<custom>/data/initial, où <custom> correspond à tout nouveau répertoire créé sous des composants respectant une structure de répertoire de type components/core.

11.8 Liaison du rôle de sécurité à l'utilisateur

Pour associer des données d'autorisation à un utilisateur, le rôle de sécurité doit être lié à l'utilisateur.

Pour ce faire, mettez à jour l'entrée de l'utilisateur spécifié dans le fichier Users.dmx situé dans le répertoire %SERVER_DIR%/components/<custom>/data/initial, où <custom> correspond à tout nouveau répertoire créé sous des composants respectant une structure de répertoire de type components/core définissant l'attribut nom de rôle sur nom de rôle comme indiqué sur la table SecurityRole.

11.9 Chargement des informations de sécurité sur la base de données

Une fois que toutes les informations ont été entrées dans les différents fichiers DMX, le gestionnaire de données doit être utilisé pour charger les données DMX sur la base de données. Pour plus d'informations, consultez le chapitre *Gestionnaire de données* du *Guide de développement de serveur Cúram*.

11.10 Création d'identificateurs de fonction

Lorsqu'une méthode est mise à la disposition du public, en définissant le stéréotype sur <<façade>>, la sécurité est automatiquement activée. Cela signifie qu'un identificateur de sécurité est automatiquement généré pour cette méthode et que l'indicateur d'activation de la sécurité de la méthode est défini sur vrai. L'identificateur de sécurité et son indicateur fidenabled sont stockés dans le fichier <ProjectName>_Fids.xml indépendant de la base de données situé dans le sous-répertoire /build/svr/gen/ddl. Ce fichier permet d'insérer les informations relatives à l'identificateur de fonction dans la base de données via le gestionnaire de données.

Un identificateur de fonction respecte la convention de nommage de type <nom de classe>.<nom de méthode> et sa longueur maximale est de 100 caractères. Par exemple, pour un objet de processus métier appelé ProductEligibility, avec deux méthodes appelées insertProduct et testProduct, deux identificateurs de fonction sont créés : ProductEligibility.insertProduct et ProductEligibility.testProduct.

Si la sécurité d'une méthode de processus est désactivée lors de la phase de conception dans le modèle, un identificateur de sécurité ou de fonction est quand même généré mais l'indicateur d'activation de la sécurité est défini sur faux. La définition de l'indicateur d'activation de la sécurité sur faux indique qu'aucune vérification d'autorisation n'est effectuée pour cette méthode.

11.11 Désactivation de la sécurité pour une méthode de processus

La définition de l'option `Générer_Sécurité` de la méthode de processus sur faux dans le modèle désactive la sécurité d'une méthode de processus.

Si la sécurité d'une méthode de processus est désactivée lors de la phase de conception d'un modèle, un identificateur de fonction est quand même généré mais l'indicateur d'activation de la sécurité est défini sur faux. La définition de l'indicateur d'activation de la sécurité sur faux signifie qu'aucune vérification d'autorisation n'est effectuée pour cette méthode.

11.12 Remarques sur la sécurité au cours du développement

Il est important d'envisager l'impact de ces options de conception lors de l'implémentation de la sécurité au cours du développement d'une application IBM Cúram Social Program Management. Elles représentent les première et dernière lignes de défense contre l'accès non autorisé à la fonctionnalité de processus applicatif. En général, la sécurité est activée pour presque toutes les méthodes de processus. La sécurité peut être désactivée pour une méthode de processus qui n'a pas besoin de la sécurité, par exemple une méthode de connexion qui est appelée lorsqu'un utilisateur tente de se connecter à une application. Tant qu'un utilisateur n'a pas été authentifié ou autorisé, il a besoin d'accéder à cette méthode pour se connecter. Il peut donc être nécessaire de désactiver la sécurité pour cette méthode.

Au cours de la phase de conception initiale d'une application, le maintien de la «synchronisation» de l'environnement de sécurité avec une application évolutive peut être fastidieux. Il est possible de désactiver la vérification d'autorisation en définissant la propriété `curam.security.disable.authorisation` dans le fichier `Application.prx`.

avertissement : Avertissement

La propriété `curam.security.disable.authorisation` doit être activée uniquement lors de la phase de conception. Elle ne doit jamais être définie sur vrai dans un environnement de production.

Enfin, il convient de noter qu'une fois que le code et les scripts ont été générés à partir d'un modèle de travail, les informations associées à un identificateur de fonction ne peuvent pas être modifiées. La modification de ces informations nécessite de modifier le modèle, puis de régénérer et reconstruire la base de données.

11.13 Contrôle de la consignation des échecs d'autorisation du client

Par défaut, les échecs d'autorisation du client Web ne sont pas enregistrés.

La propriété `curam.enable.logging.client.authcheck` contrôle si les échecs d'autorisation rencontrés par le client Web sont consignés ou non. Cette propriété a la valeur faux par défaut, ce qui signifie que ces échecs ne sont pas consignés. Lorsque la valeur est vrai, un journal de ces échecs d'autorisation est stocké sur le fichier `AuthorisationLog` de la table de base de données. Pour plus d'informations sur cette propriété, consultez la section du *Guide de développement de serveur Cúram*, `Application.prx` - Propriétés dynamiques.

11.14 Autorisation de nouveaux types d'identificateurs de sécurité

Une méthode d'interface de serveur est fournie pour permettre que l'autorisation soit appliquée directement. Cette méthode peut être ajoutée à une classe manipulant des données sur l'élément conceptuel qui est sécurisé par le nouveau type d'identificateur de sécurité.

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

Un exemple d'utilisation de cette méthode est proposé ci-dessous :

```
// The SID associated with the conceptual element
// to be secured.
String someSID = "someSID";

// Get the logged in username
String loggedUser =
    curam.util.transaction.TransactionInfo.getProgramUser();

// Check if the user has access rights
if (curam.util.security.Authorisation.isSIDAuthorised(
    someSID, loggedUser)) {
    // Do something sensitive that this user has rights to do
    ...
} else {
    // Throw an exception indicating the user doesn't have
    // access to perform this action
    AppException exception
        = new AppException(MESSAGE.ERR_USER_NO_ACCESS);
    throw exception;
}
```

Figure 7. Exemple d'utilisation d'`isSIDAuthorised()`

11.15 Analyse de la table de base de données AuthorisationLog

Tous les échecs d'autorisation sont consignés dans une table nommée `AuthenticationLog`. Voici les lignes importantes dans cette table :

Tableau 2. Contenus du journal d'authentification

Zone	Signification
Heure entrée	Horodatage de l'entrée dans le journal.
Nom d'utilisateur	Nom d'utilisateur associé à la tentative d'autorisation.
identifieurName	Identificateur de sécurité (SID) ou Identificateur fonctionnel (FID) associé à l'échec.

L'API `LogAdmin` peut être utilisée pour interroger la table de données `AuthorisationLog`. La documentation Java de cette classe doit être référencée pour plus d'informations.

Chapitre 12. Personnalisation de la cryptographie

12.1 Présentation

Ce chapitre présente comment configurer et personnaliser la cryptographie d'IBM Cúram Social Program Management.

12.2 Personnalisation du chiffrement

La modification des paramètres de chiffrement par défaut est un processus relativement simple, qui doit cependant être planifié et testé de façon adéquate. Vous devrez redémarrer une application pour rendre effectifs les changements. Selon la taille et la topologie de votre organisation et de vos déploiements, choisissez une heure à laquelle les changements en cours n'ont aucun impact. En outre, tenez compte des données (par ex., propriétés contenant des mots de passe chiffrés) gérées par Cúram Transport Manager (CTM) qui devra soit être mis à jour, soit géré pour éviter la désynchronisation des systèmes (pour plus d'informations, voir le *Guide Cúram Transport Manager*).

Pour modifier les paramètres de chiffrement par défaut, procédez comme suit :

1. Choix des nouveaux paramètres du fichier `CryptoConfig.properties` et artefacts sous-jacents - voir 4.5, «Paramètres de chiffrement Cúram», à la page 20
2. Selon les paramètres, vous aurez peut-être besoin d'effectuer des étapes supplémentaires (par ex., modification du fichier de clés en fonction 12.4, «Création d'un nouveau fichier de clés», à la page 46).
3. Modifiez le fichier `CryptoConfig.properties`. Notez que l'emplacement par défaut est le suivant : `<SERVER_DIR>/project/properties`.
4. Supprimez les fichiers `CryptoConfig.jar` (ceux-ci contiennent le fichier `CryptoConfig.properties`) et sont disponibles à l'emplacement suivant : `<JAVA_HOME>/jre/lib/ext directory ($JAVA_HOME/lib/ext sur IBM z/OS)`. Si des clients ou serveurs Cúram sont en cours d'exécution, ils devront être arrêtés pour déployer un fichier `CryptoConfig.jar` possédant les paramètres mis à jour.
5. Chiffrez à nouveau les mots de passe dans tous les fichiers de propriétés existants, comme indiqué dans 4.7, «Mots de passe chiffrés», à la page 22. Les cibles `configtest`, `configure` et `installapp` Apache Ant placent un fichier `CryptoConfig.jar` mis à jour dans le répertoire `Java lib/ext`.
6. Testez et vérifiez vos modifications.

Le test de vos modifications doit inclure la vérification des fonctionnalités susceptibles d'être impactées. Par exemple :

- Vérifiez que la cible `configtest` Ant est en cours de fonctionnement.
- Vérifiez que les programmes de traitement par lots sont en cours de fonctionnement.
- Le cas échéant, vérifiez que la cible `configure` Ant est en cours de fonctionnement.

Rubriques connexes :

- 4.6, «Paramètres de traitement Cúram», à la page 21
- 4.7, «Mots de passe chiffrés», à la page 22

12.3 Gestion des clés

La gestion de la clé confidentielle des mots de passe chiffrés Cúram s'effectue via la commande **keytool** fournie par le kit JDK ou une commande équivalente. Vous devrez prendre des décisions locales sur le placement et l'isolation de la clé confidentielle pour qui sont compatibles avec votre organisation et vos normes locales.

N'oubliez pas que des paramètres transmis à la commande **keytool** doivent être insérés dans les paramètres `CryptoConfig.properties`, qui doivent être coordonnés pour réussir le déploiement, comme décrit dans 12.2, «Personnalisation du chiffrement», à la page 45. La table suivante présente la relation entre les arguments de commande **keytool** et les propriétés de cryptographie Cúram.

Tableau 3. Relation des arguments de commande **keytool** dans les propriétés de cryptographie Cúram

Argument keytool	propriété <code>CryptoConfig.properties</code>
-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

Remarque : Le mot de passe de la clé confidentielle correspond par défaut au mot de passe `storepass` et ne doit pas être modifié.

Pour plus d'informations sur l'utilisation de la commande **keytool**, voir la documentation du kit JDK.

Rubriques connexes :

- 4.5, «Paramètres de chiffrement Cúram», à la page 20
- 4.4, «Propriétés de cryptographie», à la page 20
- 12.4, «Création d'un nouveau fichier de clés»

12.4 Création d'un nouveau fichier de clés

Pour créer un nouveau fichier de clés afin de remplacer le fichier de clés Cúram par défaut, vous devez exécuter la commande **keytool** fournie avec JDK (ou équivalent), modifier les paramètres `CryptoConfig.properties` à faire correspondre (nécessaire, uniquement si le nom et/ou l'emplacement du fichier de clés par défaut est modifié, mais le changement de nom peut rendre vos personnalisations plus évidentes) et vérifier que les cibles Curam Ant peuvent trouver le nouveau fichier de clés (nécessaire uniquement si l'emplacement par défaut est modifié).

Par exemple :

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

La section 12.3, «Gestion des clés», à la page 45 identifie les arguments de commande **keytool** associés aux paramètres `CryptoConfig.properties`.

L'emplacement par défaut du fichier de clés est le répertoire `<SERVER_DIR>/project/properties`, qui contient une structure de sous-répertoire rappelant le kit utilisé : «`ibm`» pour le kit JDK IBM et «`sun`» pour le kit JDK Oracle. Lors de la création d'un fichier de clés, les scripts de génération Curam s'attendent à le trouver dans le dossier du kit JDK IBM : `<SERVER_DIR>/project/properties/ibm`. Si vous souhaitez utiliser un emplacement différent de l'emplacement par défaut, vous pouvez effectuer l'une des deux actions suivantes :

1. Utilisez un emplacement absolu pour le fichier de clés, comme décrit dans 4.4, «Propriétés de cryptographie», à la page 20. Dans ce dossier, les fichiers de clés par défaut Curam qui se trouvent dans `CryptoConfig.jar` sont ignorés en faveur du paramètre absolu `CryptoConfig.properties`.

2. Utilisez la propriété `Ant crypto.prop.file.location` lorsque vous exécutez une cible décrite dans 12.2, «Personnalisation du chiffrement», à la page 45, qui permet de créer et de copier le fichier `CryptoConfig.jar` pour désigner l'autre emplacement. L'emplacement spécifié doit indiquer la structure de votre kit JDK - «ibm» ou «sun». Par exemple :
 - Placez le nouveau fichier de clés dans un emplacement tel que celui de Windows pour le kit JDK IBM : `C:\Curam\keystore\ibm\MyOrganization.keystore`
 - Désignez cet emplacement lors de l'exécution des cibles de génération : `ant configure -Dcrypto.prop.file.location=C:\Curam\keystore`

Remarque : Dans l'exemple ci-dessous, le changement du nom du fichier de clés en `MyOrganization.keystore` nécessite un changement correspondant en `CryptoConfig.properties` en fonction de 4.4, «Propriétés de cryptographie», à la page 20.

Suivant la création du fichier de clés, vous devez suivre les étapes contenues dans 12.2, «Personnalisation du chiffrement», à la page 45.

Rubriques connexes :

- 12.3, «Gestion des clés», à la page 45
- 12.2, «Personnalisation du chiffrement», à la page 45

12.5 Personnalisation du traitement

La modification des paramètres de traitement par défaut est un processus relativement simple, qui doit cependant être planifié et testé de façon adéquate. Vous devrez redémarrer une application pour rendre effectifs les changements. Selon la taille et la topologie de votre organisation et de vos déploiements, choisissez une heure à laquelle les changements en cours n'ont aucun impact. En outre, tenez compte des données (par ex., mots de passe utilisateur) gérées par *Cúram Transport Manager (CTM)* qui devront soit être mises à jour, soit gérées pour éviter la désynchronisation des systèmes (pour plus d'informations, voir le *Guide Cúram Transport Manager*).

Le processus est présenté en détail dans 12.7, «Utilisation des paramètres de traitement obsolètes pour une période de migration», à la page 48.

Rubriques connexes :

- 4.6, «Paramètres de traitement Cúram», à la page 21
- 12.6, «Spécification d'un sel de traitement»

12.6 Spécification d'un sel de traitement

Bien que *Cúram* ne spécifie pas de sel de traitement prêt à l'emploi, la spécification d'un sel pour mots de passe traités permet d'obtenir un niveau de protection supplémentaire contre les attaques en force brute.

Pour spécifier un sel pour les mots de passe chiffrés :

1. Choisissez une chaîne aléatoire suffisamment longue.
2. Chiffrez cette chaîne à l'aide de la cible **encrypt** (voir le *Guide de développement du serveur Cúram* .
3. Placez la chaîne chiffrée dans un fichier.
4. Spécifiez l'emplacement du fichier contenant la chaîne de sel chiffré à l'aide de la propriété `curam.security.crypto.digest.salt.location` dans `CryptoConfig.properties` et assurez-vous que n'importe quel fichier `CryptoConfig.jar` déployé reflète les paramètres mis à jour.

A des fins de maniabilité, effectuez ces changements avec les étapes incluses dans 12.7, «Utilisation des paramètres de traitement obsolètes pour une période de migration», à la page 48.

12.7 Utilisation des paramètres de traitement obsolètes pour une période de migration

L'utilisation de paramètres de traitement obsolètes signifie que vous migrez les mots de passe chiffrés existants vers une nouvelle configuration de cryptographie (par ex., new sel) et souhaitez migrer automatiquement les mots de passe utilisateur Cúram pour une période de temps. Cela s'applique aux utilisateurs Cúram internes et externes, mais pas aux utilisateur gérés par des systèmes de sécurité tiers, tels que le protocole LDAP.

La procédure à effectuer est la suivante :

1. Choisissez un moment pendant lequel votre système Cúram peut être arrêté.
2. Copiez les noms de propriété et les valeurs de traitement qui existent dans `CryptoConfig.properties` et renommez les propriétés en indiquant les nouveaux noms de propriété obsolètes.
3. Modifiez les noms de propriété de traitement existants dans le fichier `CryptoConfig.properties`.
4. Définissez la propriété `curam.security.convertsupersededpassworddigests.enabled` sur 'true'.
5. Définissez la propriété `curam.security.crypto.upgrade.start` pour vous aider à effectuer le suivi lorsque vous avez introduit la configuration mise à jour. Cette valeur peut être utilisée ci-dessous pour vous aider à gérer les mots de passe utilisateur non migrés.
6. Redémarrez le serveur d'applications. Toutefois, notez les points suivants.

Remarque : L'utilisateur de services Web par défaut Cúram (WEBSVCS) ou un utilisateur non traité via `CuramLoginModule` n'est pas disponible pour une migration de mot de passe automatique. Vous devez réinitialiser ces utilisateurs avant de redémarrer le serveur d'applications. Pour ce faire :

1. Obtenez la nouvelle valeur de mot de passe de traitement via la cible de traitement Ant (par ex., `ant digest -Dpassword=password`).
2. Mettez à jour la valeur du mot de passe dans la base de données, qui peut s'effectuer facilement via SQL (par ex., `UPDATE USERS SET PASSWORD='<new digest value>' WHERE USERNAME='WEBSVCS';`).
3. Vous pouvez démarrer maintenant le serveur d'applications

Une fois la période de migration terminée, définissez la propriété `curam.security.convertsupersededpassworddigests.enabled` sur 'false' et annulez la définition de la propriété `curam.security.crypto.upgrade.start` property.

Les utilisateurs qui ne se sont pas connectés lors de la période de migration ne parviendront plus maintenant à se connecter en raison de mots de passe non concordants. Il existe deux manières de traiter les mots de passe non mis à jour lors de la période de migration :

1. Demandez à ces utilisateurs de contacter votre support interne pour réinitialiser les mots de passe via l'interface utilisateur d'administration.
2. Identifiez manuellement les utilisateurs dans la table `USERS` de Cúram qui n'ont pas été mis à jour pendant la période de migration et définissez le nouveau mot de passe par défaut via SQL (voir la cible **digest** décrite dans le *Guide de développement de serveur Cúram* pour obtenir les nouvelles valeurs de mot de passe de traitement) ou via les écrans d'utilisateur d'administration. Par exemple, utilisez la requête suivante : `SELECT username FROM users WHERE lastwritten between timestamp('2013-06-01 15:00:00') AND timestamp('2013-09-01 00:00:00')`

Ne laissez `curam.security.convertsupersededpassworddigests.enabled` défini sur true indéfiniment pour les raisons suivantes :

1. Il est inutile d'effectuer la mise à niveau de la configuration 'A' vers la configuration 'B' et de maintenir active la configuration 'A' d'origine.
2. Les paramètres de cryptographie actifs dans le système sont plus faibles.

3. Pour utiliser cette fonctionnalité lors d'une mise à niveau ultérieure (par ex., de la configuration 'B' à la configuration 'C', tous les mots de passe de la configuration 'A' doivent au moins être mis à niveau vers la configuration 'B'.

Remarque : Des fichiers (par ex., DMX) avec traitements stockés doivent être pris en compte en fonction de votre stratégie de migration pour indiquer les valeurs correctes.

Remarque : Toute utilisation de Cúram Transport Manager (CTM) lors d'une migration doit être prise en compte pour garantir la compatibilité des paramètres et des attentes entre les système source et cible.

Rubriques connexes :

- 4.5, «Paramètres de chiffrement Cúram», à la page 20
- 4.6, «Paramètres de traitement Cúram», à la page 21

12.8 Modification de votre configuration de cryptographie pour un système de production

Bien que les paramètres de cryptographie prêts à l'emploi soient adaptés pour les environnements de développement ou de test typiques, ils doivent être modifiés pour les environnements de production à des fins de protection et d'isolation entre ces environnements relativement peu risqués et les environnements très risqués.

Des changements typiques apportées à la configuration de cryptographie prête à l'emploi en préparation de la production, peuvent inclure :

- Attribution d'une nouvelle clé confidentielle.
 - Ce type de clé peut être généré à l'aide de l'utilitaire keytool du kit JDK. Voir 12.4, «Création d'un nouveau fichier de clés», à la page 46
 - Cette clé confidentielle doit être stocké dans un fichier de clés séparé.
 - Les propriétés de ces changements de clé confidentielle sont celles décrites dans 12.3, «Gestion des clés», à la page 45.
- Attribution de nouveaux paramètres de traitement
 - Les nouveaux paramètres de traitement peuvent inclure un nouveau sel de cryptage, un compteur d'itérations et/ou un algorithme.
 - Les propriétés de ces changements de traitement sont celles décrites dans 4.6, «Paramètres de traitement Cúram», à la page 21 et 12.6, «Spécification d'un sel de traitement», à la page 47 et dans le processus présenté dans 12.7, «Utilisation des paramètres de traitement obsolètes pour une période de migration», à la page 48.

N'oubliez pas de maintenir vos fichiers de configuration isolés du personnel qui n'a pas absolument besoin d'y accéder. Plus particulièrement, maintenez les informations de développement, de test et de configuration de la production isolés.

Chapitre 13. Personnalisation des applications d'utilisateurs externes

13.1 Présentation

Etant donné que les utilisateurs externes sont traités différemment des utilisateurs internes, une application IBM Cúram Social Program Management est obligatoire spécifiquement pour les utilisateurs externes.

13.2 Création d'une application d'utilisateur externe

Une nouvelle application client Web doit être développée pour les utilisateurs externes. Pour plus d'informations sur la création d'une nouvelle application client Web, consultez le *Manuel de référence du client Web Cúram*.

13.3 Création d'une page de connexion client d'utilisateur externe

Une nouvelle page `logon.jsp` doit être créée pour une application d'utilisateur externe. L'application Social Program Management Platform expédie une page de connexion par défaut, `logon.jsp`, située dans le répertoire `lib/curam/web/jsp` de l'environnement CDEJ (Client Development Environment for Java). Le fichier doit être copié dans un dossier `webclient/components/<custom component>/WebContent` de l'application de client Web et doit être modifié comme suit :

L'élément `table` doit être étendu afin d'inclure une zone de saisie masquée `user_type` :

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

Où `EXTERNE` indique le type d'utilisateur externe. Celui-ci peut être défini sur n'importe quelle valeur, excepté `INTERNE`.

13.4 Création d'une page de connexion automatique de client de l'utilisateur externe

Certaines applications client d'utilisateur externe ne nécessitent pas d'authentification de l'utilisateur et aucun nom d'utilisateur ni mot passe n'est donc requis. Il est impossible de désactiver l'authentification dans IBM Cúram Social Program Management, aussi le meilleur moyen de satisfaire cette exigence consiste à écrire un script de connexion automatique.

Le script de connexion automatique utilise un nom d'utilisateur codé en dur et un mot de passe et les fournit en tant qu'informations d'authentification si nécessaire. Cela signifie que tous les utilisateurs d'une telle application s'exécutent toujours sous le même nom d'utilisateur. L'utilisation de ce script doit être limitée aux véritables applications d'accès ouvert.

Lors de l'implémentation d'applications nécessitant une connexion automatique, il convient d'envisager les implications de la gestion de session. La gestion de session dans IBM Cúram Social Program Management conserve les informations de session d'un utilisateur pour s'assurer que lorsque l'utilisateur se connecte à nouveau, les informations de session pertinentes, à savoir les onglets et la navigation, s'ouvrent là où il les a laissées. Si un utilisateur s'est connecté automatiquement, ces informations ne doivent pas être conservées, par conséquent la gestion de session peut devoir être désactivée dans ce scénario. Pour plus d'informations sur la désactivation de la gestion de session, consultez le *Manuel de référence du client Web Cúram*.

Voici des exemples de scripts JSP de connexion et de déconnexion automatiques.

Remarque : Les implémentations et les configurations de sécurité diffèrent entre les fournisseurs de serveurs d'applications, ces exemples risquent donc de ne pas fonctionner dans tous les cas ou pour toutes les versions de serveurs d'applications.

```
<?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>

  <!-- Réacheminement automatique vers la vérification de la sécurité de connexion des
    détails de l'utilisateur indiqués ci-dessous -->

  <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="EXTERNE" />
      </form>
    </body>
  </html>
</jsp:root>
```

Figure 8. JSP de connexion automatique


```

<?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>

```

Figure 9. JSP de déconnexion automatique

13.5 Extension de la classe d'utilisateurs d'accès public

Pour «ancrer» la solution personnalisée dans l'application, la classe abstraite `curam.util.security.PublicAccessUser` doit être étendue. Cela nécessite d'implémenter l'interface `curam.util.security.ExternalAccessSecurity`. Cette classe concrète sera utilisée lors des processus d'authentification et d'autorisation pour déterminer les informations requises liées à l'utilisateur externe. Cette classe et ses méthodes sont décrites en détail ci-après.

13.6 Authentification d'un utilisateur externe

La méthode `authenticate()` est responsable de l'authentification d'un utilisateur externe. Elle est appelée au cours du processus d'authentification si l'utilisateur est identifié en tant qu'utilisateur externe. En cas d'utilisateurs externes, cette méthode est appelée à la place de l'authentification configurée.

Remarque : Si un autre mécanisme d'authentification, par exemple LDAP, est configuré, les utilisateurs externes doivent être en mesure de s'authentifier par rapport à ce mécanisme.

```

/**
 * L'implémentation de cette méthode doit valider l'identificateur
 * et le mot de passe, puis renvoyer le résultat de la validation. Si les informations sont
 * valides, le code de table de codes SecurityStatus.LOGIN doit être renvoyé.
 *
 * @param identifieur Identificateur de l'utilisateur externe.
 * @param password Mot de passe en tant que matrice de caractères.
 * @param userType Type d'utilisateur externe.
 *
 * @return Statut de l'authentification sous la forme d'un code de table de codes.
 *
 * @throws AppException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */

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

```

Les paramètres d'entrée de la méthode incluent un identificateur, le mot de passe traité comme une matrice de caractères et le type d'utilisateur externe à authentifier.

Le paramètre `userType` vise à permettre la prise en charge de plusieurs types d'utilisateur qui nécessitent différents mécanismes d'authentification. L'utilisation de ce paramètre dépend de l'implémentation personnalisée.

Le résultat prévu de cette méthode est une entrée provenant de la table de codes `curam.util.codetable.SECURITYSTATUS`. En cas d'authentification réussie, le résultat doit être :

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

En cas d'échec d'authentification, cette table de codes contient un certain nombre d'entrées, notamment `BADUSER`, `BADPWD` et `PWDEXPIRED`. Cette table de codes peut être étendue pour inclure des codes personnalisés comme indiqué dans le *Cúram Server Guide de développement du serveur Cúram*.

Le résultat d'authentification renvoyé par cette méthode est automatiquement consigné dans la table de base de données `AuthenticationLog`. Pour plus d'informations sur cette table, consultez le *Guide de développement de serveur Cúram*.

La classe abstraite `PublicAccessUser` définit également les méthodes abstraites suivantes que des sous-classes concrètes doivent implémenter :

- La méthode `upgradeSafePasswordValidation()` est nécessaire pour autoriser la comparaison de mot de passe et est définie comme suit :

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

```

- La méthode `setPassword()` consiste à autoriser l'implémenteur à conserver le mot de passe (par ex., nouveau mot de passe) dans le cas de mises à niveau de cryptographie. Cette méthode est appelée lorsque la méthode `upgradeSafePasswordValidation()` est appelée. Voici la définition de la méthode :

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

```

Voir la documentation Java associée de la classe `PublicAccessUser` pour plus d'informations sur les méthodes ci-dessus.

13.7 Détermination des détails de l'utilisateur externe

Les détails d'un utilisateur externe sont extraits en appelant la méthode `getLoginDetails()` de l'interface `curam.util.security.ExternalAccessSecurity`. Ces détails sont renvoyés directement après l'authentification pour diriger l'utilisateur externe vers la page d'accueil appropriée.

```
/**
 * L'implémentation de cette méthode doit récupérer les
 * détails de l'utilisateur obligatoires pour le rediriger vers la page
 * d'application appropriée. Ces informations incluent le nom de
 * la page d'accueil de l'application pour l'utilisateur, l'environnement local par défaut pour
 * l'utilisateur ainsi qu'une liste d'avertissements/messages destinés à l'utilisateur.
 *
 * @param identifiant Identificateur de l'utilisateur externe.
 *
 * @return Détails de l'utilisateur, y compris la page
 *         d'accueil de l'application.
 *
 * @throws ApplicationException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */
UserLoginDetails getLoginDetails (Identificateur de chaîne)
    throws ApplicationException, InformationalException;
```

Une instance de la classe `curam.util.security.UserLoginDetails` doit être créée et renvoyée à partir de cette méthode. Les informations suivantes doivent être renvoyées à l'aide de cette classe :

- `UserLoginDetails.setApplicationCode (Code de chaîne)`
Code correspondant à la page d'accueil de l'application pour l'utilisateur externe.
Il doit s'agir d'une entrée valide dans la table de codes `APPLICATION_CODE`.
- `UserLoginDetails.setDefaultLocale (Environnement local par défaut de la chaîne)`
Environnement local par défaut pour l'utilisateur externe.
Il s'agit de l'environnement local dans lequel l'application s'affiche par défaut pour l'utilisateur externe.
- `UserLoginDetails.addInformationals(InformationalManager informationalManager)`
Informations qui doivent s'afficher pour l'utilisateur externe.
La classe `curam.util.exception.InformationalManager` peut être utilisée pour créer un certain nombre de messages d'information ou d'avertissement qui s'affichent lorsque l'utilisateur externe se connecte. Cela peut être par exemple un avertissement indiquant à l'utilisateur externe que son mot de passe arrive à expiration.

13.8 Autorisation d'un utilisateur externe

La méthode `getSecurityRole()` est utilisée dans le cadre de l'autorisation pour déterminer le rôle de sécurité associé à l'utilisateur externe. Les rôles de sécurité utilisés pour les utilisateurs externes sont configurés de la même manière que les rôles de sécurité destinés aux utilisateurs internes.

```

/**
 * L'implémentation de cette méthode doit renvoyer le rôle
 * de sécurité associé à l'utilisateur externe à des fins
 * d'autorisation. Si l'utilisateur n'existe pas, la valeur NULL doit être
 * renvoyée.
 *
 * @param identifiant Identificateur de l'utilisateur externe.
 *
 * @return Rôle de sécurité pour l'autorisation.
 *
 * @throws ApplicationException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */
String getSecurityRole (Identificateur de chaîne)
    throws ApplicationException, InformationalException;

```

L'environnement CDEJ fait appel à une implémentation de cette méthode dans le cadre du processus d'autorisation si l'utilisateur n'existe pas dans le cache de sécurité. Seuls les utilisateurs internes peuvent exister dans le cache de sécurité. Cela signifie que les identificateurs utilisés pour identifier les utilisateurs externes doivent être uniques et ne doivent pas entrer en conflit avec les noms d'utilisateurs configurés pour les utilisateurs internes, à moins que l'interface `UserScope` personnalisée décrite dans 7.3, «Portée d'utilisateur», à la page 29 soit implémentée. Sinon, en cas de conflit de nom d'utilisateur, les droits d'accès affectés à l'utilisateur interne sont également utilisés pour l'utilisateur externe.

Si un rôle ne peut pas être déterminé pour l'utilisateur externe, la valeur NULL doit être renvoyée afin que l'environnement CDEJ puisse consigner l'erreur d'autorisation correctement.

13.9 Détermination du type d'utilisateur

La méthode `getUserType()` permet de déterminer si un utilisateur est un utilisateur externe.

```

/**
 * Renvoyez le type d'utilisateur. Cela permet la prise en charge de
 * différents types d'utilisateurs externes. S'il n'y a qu'un seul
 * type d'utilisateur externe, renvoyez simplement "EXTERNE".
 *
 * @param identifiant Identificateur de l'utilisateur externe.
 *
 * @return Type d'utilisateur externe.
 *
 * @throws ApplicationException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */
String getUserType (Identificateur de chaîne final)
    throws ApplicationException, InformationalException;

```

La valeur `getProgramUserType()` dans `curam.util.transaction.TransactionInfo` demande à cette méthode de renvoyer le type d'utilisateur si ce dernier n'est pas reconnu en tant qu'utilisateur interne. Pour les utilisateurs internes, la valeur «INTERNE» est toujours renvoyée.

Pour les utilisateurs externes, il peut exister plusieurs types d'utilisateurs externes ; cette méthode doit donc renvoyer le type spécifique d'utilisateur externe.

13.10 Prévention de la suppression d'un rôle de sécurité : Nombre d'utilisations du rôle

La méthode `getRoleUsageCount()` permet d'empêcher la suppression d'un rôle de sécurité actuellement référencé par un utilisateur externe.

```

/**
 * Renvoiez le nombre d'utilisateurs utilisant un rôle particulier. Cette
 * méthode permet de s'assurer qu'un rôle ne peut pas être supprimé lorsqu'il
 * est utilisé par un utilisateur externe.
 *
 * @param role Nom du rôle de sécurité.
 *
 * @return Nombre d'utilisateurs utilisant actuellement le
 *         rôle spécifié.
 *
 * @throws ApplicationException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */
int getRoleUsageCount (Rôle de chaîne)
    throws ApplicationException, InformationalException;

```

Les rôles de sécurité référencés par un utilisateur, interne ou externe, ne peuvent pas être supprimés. Cette méthode doit renvoyer un nombre de 1 minimum si des utilisateurs externes font référence au rôle spécifié.

13.11 Récupération d'un nom d'utilisateur enregistré

La méthode `getRegisteredUserName()` permet de récupérer le nom d'utilisateur de dossier correct, qui peut être indépendant du nom d'utilisateur entré lors de la connexion.

```

/**
 * Obtient le dossier correct pour cet utilisateur, indépendamment du dossier
 * mixte qui a peut-être été entré par l'utilisateur connecté.
 *
 * @param identifiant Identificateur de l'utilisateur externe,
 * dont le dossier ne correspond peut-être pas à l'identificateur conservé
 * pour l'utilisateur.
 *
 * @return Dossier réel de cet utilisateur, avant que son dossier
 * n'ait été modifié par des facteurs externes.
 *
 * @throws ApplicationException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */
public String getRegisteredUserName (Identificateur de chaîne final)
    throws ApplicationException, InformationalException;

```

L'implémentation par défaut de cette méthode doit renvoyer le nom d'utilisateur qui a été fourni. Cette méthode peut nécessiter le changement du dossier du nom d'utilisateur renvoyé uniquement si la propriété `curam.security.casesensitive` est définie sur `Faux`.

Remarque : Lorsque la propriété `curam.security.casesensitive` est définie sur `Faux` et est obligatoire pour les utilisateurs externes, toutes les méthodes de cette interface doivent traiter toutes les exigences spécifiques du dossier.

13.12 Relevé des préférences utilisateur

La méthode `getUserPreferenceSetID()` permet de récupérer l'ID d'ensemble de préférences utilisateur associé à un utilisateur externe. S'il n'existe pas de préférences utilisateur pour un utilisateur externe, les préférences par défaut sont utilisées pour celui-ci. Pour plus d'informations sur les préférences utilisateur, consultez le chapitre *Préférences utilisateur* du *Guide de développement de serveur Cúram*.

```

/**
 * Cette méthode permet de récupérer un ensemble de préférences utilisateur
 * associé à un utilisateur externe. La valeur userPrefSetID est une
 * clé externe de la table UserPreferenceInfo.
 * La table UserPreferenceInfo contient des informations sur
 * les préférences utilisateur.
 *
 * @param identifier Identificateur de l'utilisateur externe.
 *
 * @return Valeur userPrefSetID pour l'utilisateur externe.
 *
 * @throws AppException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */
String getUserPreferenceSetID (Identificateur de chaîne final)
    throws AppException, InformationalException;

```

L'implémentation par défaut de cette méthode doit renvoyer l'ID d'ensemble de préférences utilisateur associé à un utilisateur externe.

13.13 Modification des préférences utilisateur

La méthode `modifyUserPreferenceSetID()` permet de mettre à jour les détails de l'utilisateur externe avec un nouvel ensemble de préférences utilisateur. Pour plus d'informations sur les préférences utilisateur, consultez le chapitre Préférences utilisateur.

```

/**
 * Cette méthode met à jour les détails de l'utilisateur externe avec de nouvelles
 * préférences utilisateur.
 *
 * @param userPreferenceSetID ID des préférences utilisateur.
 * @param username Identificateur de l'utilisateur externe.
 *
 * @throws AppException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */
void modifyUserPreferenceSetID(
    final String userPreferenceSetID, final String username)
    throws AppException, InformationalException;

```

L'implémentation par défaut de cette méthode doit mettre à jour l'ID d'ensemble de préférences utilisateur associé à un utilisateur externe.

13.14 Configuration de la sécurité d'accès externe

La propriété `curam.custom.externalaccess.implementation` doit être définie dans le fichier `Application.prx` pour indiquer le nom qualifié complet de la classe qui implémente l'interface ci-dessus.

Remarque : La propriété `curam.custom.externalaccess.implementation` n'est pas dynamique et si on la modifie, l'application doit être redémarrée avant que la modification soit appliquée.

13.15 Détermination d'un utilisateur interne ou externe à l'aide de l'interface `UserScope`

L'interface personnalisée `UserScope` est disponible pour prendre en charge les autres méthodes de détermination d'un utilisateur interne ou externe. Par exemple, cette interface personnalisée peut être implémentée pour déterminer le type d'utilisateur en fonction des informations supplémentaires, et supprimer l'exigence de noms uniques entre les utilisateurs externes et internes.

Pour fournir une implémentation personnalisée permettant de déterminer le type d'utilisateur, l'interface `curam.util.security.UserScope` doit être implémentée. Cette interface possède une méthode `isUserExternal()` qui détermine le type d'utilisateur. Cette méthode doit renvoyer la valeur `Vrai` si l'utilisateur est considéré comme externe ou `Faux` s'il est interne.

Pour indiquer l'implémentation personnalisée à utiliser, la propriété `curam.custom.userscope.implementation` doit être définie dans le fichier `Application.prx`. Cette valeur doit être définie sur le nom qualifié complet de la classe qui implémente l'interface `UserScope`.

Remarque : La propriété `curam.custom.userscope.implementation` n'est pas dynamique et si on la modifie, l'application doit être redémarrée avant que la modification soit appliquée.

La méthode `isUserExternal()` de l'interface `UserScope` est indiquée en détail ci-après :

13.16 Détermination du type d'utilisateur

La méthode `isUserExternal()` est appelée à n'importe quel endroit de l'application où le type d'utilisateur doit être déterminé. Cela inclut le moment où l'utilisateur se connecte à l'application et celui où il tente une autorisation afin d'accéder aux éléments sécurisés IBM Cúram Social Program Management.

```
/**
 * L'implémentation de cette méthode doit déterminer le type
 * d'utilisateur qui est connecté à l'application. Il existe 2 types
 * d'utilisateurs : INTERNE et EXTERNE. Si l'utilisateur est un utilisateur EXTERNE,
 * alors cette méthode doit renvoyer la valeur Vrai. Si la valeur Faux est renvoyée,
 * alors l'utilisateur est considéré comme INTERNE.
 *
 * @param username - Nom d'utilisateur.
 * @return Valeur booléenne de type Vrai indiquant un utilisateur EXTERNE,
 * Faux indiquant un utilisateur INTERNE.
 *
 * @throws AppException Signature de l'exception générique.
 * @throws InformationalException Signature de l'exception générique.
 */
boolean isUserExternal(Nom d'utilisateur de chaîne)
    throws AppException, InformationalException;
```

Remarques

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM. IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Pour le Canada, veuillez adresser votre courrier à :

IBM Director of Commercial Relations
IBM Canada Ltd.
3600 Steeles Avenue East
Markham, Ontario
L3R 9Z7
Canada

Les informations sur les licences concernant les produits utilisant un jeu de caractères double octet peuvent être obtenues par écrit à l'adresse suivante :

Licence sur la propriété intellectuelle
Mentions légales et droit de propriété intellectuelle
IBM Japon Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japon

Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales. INTERNATIONAL BUSINESS MACHINES CORPORATION FOURNIT CETTE PUBLICATION "EN L'ETAT" SANS GARANTIE D'AUCUNE SORTE, EXPLICITE OU IMPLICITE, Y COMPRIS NOTAMMENT, LES GARANTIES IMPLICITES DE NON-CONTREFACON, DE QUALITE MARCHANDE OU D'ADEQUATION A UN USAGE PARTICULIER. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies. Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

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

Pour le Canada, veuillez adresser votre courrier à :
IBM Director of Commercial Relations
IBM Canada Ltd
3600 Steeles Avenue East
Markham, Ontario
L3R 9Z7 Canada

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans ce document et tous les éléments sous licence disponibles s'y rapportant sont fournis par IBM, conformément aux dispositions du Livret contractuel, des Conditions Internationales d'Utilisation de Logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles.

IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Tous les tarifs indiqués sont les prix de vente actuels suggérés par IBM et sont susceptibles d'être modifiés sans préavis. Les tarifs appliqués peuvent varier selon les revendeurs.

Ces informations sont fournies uniquement à titre de planification. Elles sont susceptibles d'être modifiées avant la mise à disposition des produits décrits.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

LICENCE DE COPYRIGHT :

Le présent logiciel contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de

copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes. Les exemples de programmes sont fournis "EN L'ÉTAT", sans garantie d'aucune sorte. IBM décline toute responsabilité relative aux dommages éventuels résultant de l'utilisation de ces exemples de programmes.

Toute copie intégrale ou partielle de ces exemples de programmes et des oeuvres qui en sont dérivées doit inclure une mention de droits d'auteur libellée comme suit :

© (nom de votre société) (année). Des segments de code sont dérivés des exemples de programmes d'IBM Corp.

© Copyright IBM Corp. _entrez l'année ou les années_. Tous droits réservés.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

Documentation sur l'interface de programmation

Cette publication décrit les interfaces de programmation prévues qui permettent au client d'écrire des programmes afin d'obtenir les services d'IBM Cúram Social Program Management.

Marques

IBM, le logo IBM et [ibm.com](http://www.ibm.com) sont des marques ou des marques déposées d'International Business Machines Corp. dans de nombreux pays. Les autres noms de produit et de service peuvent être des marques d'IBM ou d'autres sociétés. La liste actualisée des marques IBM est disponible sur la page Web "Copyright and trademark information" à l'adresse suivante : <http://www.ibm.com/legal/us/en/copytrade.shtml>.

Apache est une marque d'Apache Software Foundation.

Oracle, WebLogic Server, Java ainsi que tous les logos et toutes les marques Java sont des marques d'Oracle et/ou de ses affiliés.

D'autres sociétés sont propriétaires des autres marques qui pourraient apparaître dans ce document. Les autres noms de sociétés, de produits et de services peuvent appartenir à des tiers.

