HITACHI
Inspire the Next

Cosminexus V11 アプリケーションサーバ 機能解説 セキュリティ管理機能編

解説書

3021-3-J09-30

前書き

■ 対象製品

マニュアル「アプリケーションサーバ & BPM/ESB 基盤 概説」の前書きの対象製品の説明を参照してください。

■ 輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

HITACHI，Cosminexus，DABroker，HiRDB，JP1，Keymate，OpenTP1，TPBroker，uCosminexus は，株式会社 日立製作所の商標または登録商標です。

Active Directory は，マイクロソフト 企業グループの商標です。

IBM は，世界の多くの国で登録された International Business Machines Corporation の商標です。

Microsoft は，マイクロソフト 企業グループの商標です。

Oracle および Java は，オラクルおよびその関連会社の登録商標です。

Red Hat is a registered trademark of Red Hat, Inc. in the United States and other countries.

Red Hat は，米国およびその他の国における Red Hat, Inc.の登録商標です。

Red Hat Enterprise Linux is a registered trademark of Red Hat, Inc. in the United States and other countries.

Red Hat Enterprise Linux は，米国およびその他の国における Red Hat, Inc.の登録商標です。

SQL Server は，マイクロソフト 企業グループの商標です。

Tivoli は，世界の多くの国で登録された International Business Machines Corporation の商標です。

UNIX は，The Open Group の登録商標です。

Windows は，マイクロソフト 企業グループの商標です。

Windows Server は，マイクロソフト 企業グループの商標です。

その他記載の会社名，製品名などは，それぞれの会社の商標もしくは登録商標です。

Eclipse は，開発ツールプロバイダのオープンコミュニティである Eclipse Foundation, Inc.により構築された開発ツール統合のためのオープンプラットフォームです。

Eclipse Ready ロゴは，Eclipse Foundation,Inc.の米国における商標です。

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

Portions of this software were developed at the National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign.

This product includes software developed by the University of California, Berkeley and its contributors.

This software contains code derived from the RSA Data Security Inc. MD5 Message-Digest Algorithm, including various modifications by Spyglass Inc., Carnegie Mellon University, and Bell Communications Research, Inc (Bellcore).

Regular expression support is provided by the PCRE library package, which is open source software, written by Philip Hazel, and copyright by the University of Cambridge, England. The original software is available from ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/

This product includes software developed by Ralf S. Engelschall <rse@engelschall.com> for use in the mod_ssl project (http://www.modssl.org/).

■ 発行

2022 年 3 月 3021-3-J09-30

■ 著作権

All Rights Reserved. Copyright (C) 2020, 2022, Hitachi, Ltd.

変更内容

変更内容(3021-3-J09-30) uCosminexus Application Server 11-20，uCosminexus Client 11-20，uCosminexus Developer 11-20，uCosminexus Service Architect 11-20，uCosminexus Service Platform 11-20

追加・変更内容	変更個所
Java EE 8 に対応した。	1.1
アプリケーションサーバが対応している標準仕様に，JAX-RS 2.1，JPA 2.2，JSON-B 1.0 を追加した。	1.1.3
アプリケーションサーバ 11-20 での主な機能変更についての説明を追加した。	1.4
SSL についてのメッセージにエラーレベルを追加した。	19.8

単なる誤字・脱字などはお断りなく訂正しました。

はじめに

このマニュアルをお読みになる際の前提情報については，マニュアル「アプリケーションサーバ & BPM/ESB 基盤 概説」のはじめにの説明を参照してください。

目次

前書き　2
変更内容　4
はじめに　5

第 1 編　概要

1　アプリケーションサーバの機能　19
1.1　機能の分類　20
1.1.1　アプリケーションの実行基盤としての機能　22
1.1.2　アプリケーションの実行基盤を運用・保守するための機能　23
1.1.3　機能とマニュアルの対応　24
1.2　システムの目的と機能の対応　27
1.2.1　認証のための機能　27
1.2.2　暗号化のための機能　28
1.2.3　不正処理の実行防止のための機能　29
1.2.4　そのほかの機能　29
1.3　このマニュアルに記載している機能の説明　30
1.3.1　分類の意味　30
1.3.2　分類を示す表の例　30
1.4　アプリケーションサーバ 11-20 での主な機能変更　32
1.4.1　標準機能・既存機能への対応　32

2　アプリケーションサーバでのセキュリティ管理　33
2.1　この章の構成　34
2.2　セキュリティを確保するための観点　35
2.2.1　セキュリティを確保するためのシステム構成を検討する　35
2.2.2　システムをセキュアに運用する　35
2.2.3　不正なユーザからのアクセスを防止する（認証機能）　36
2.2.4　通信路でのセキュリティを確保する（暗号化機能）　36
2.2.5　不正な処理の実行を防止する　36
2.2.6　そのほかの観点を確認する　37
2.3　セキュリティを確保する方法および機能の詳細　38
2.4　セキュリティを確保する方法および機能を使用する場合の注意事項　39
2.4.1　証明書について　39

第 2 編　システム設計

3　セキュリティを確保するためのシステム構成　40

3.1　この章の構成　41
3.2　ファイアウォールを使用する構成を検討する　42
3.2.1　サーブレットと JSP に対するファイアウォールの配置　42
3.2.2　Session Bean と Entity Bean に対するファイアウォールの配置　43
3.2.3　リソースマネジャに対するファイアウォールの配置　44
3.3　DMZ へのリバースプロキシの配置を検討する　46
3.3.1　リバースプロキシの配置　46

4　セキュアなシステムの検討　50

4.1　この章の構成　51
4.2　セキュアなシステムの検討の概要　52
4.3　セキュアなシステムの構成の検討　54
4.4　システムの使用者の検討　56
4.5　システムが扱う資産の検討　57
4.6　セキュアなシステムの前提条件の確認　58
4.6.1　物理的な前提条件　58
4.6.2　運用上の前提条件　58
4.7　想定される脅威の分析　60
4.8　対策の検討　61
4.8.1　前提条件に対して実施する対策　61
4.8.2　想定した脅威に対して実施する対策　62
4.8.3　対策を実施したセキュアなシステムの動作　64
4.9　作業手順の検討　67
4.9.1　作成する作業手順書の概要　67
4.9.2　システムの構築手順の検討　68
4.9.3　システムの再構築手順の検討　75
4.9.4　システムの運用手順の検討　77
4.10　システムの監査方法の確認　80
4.10.1　監査ログの入手　80
4.10.2　監査ログの調査　80
4.11　外部ネットワークを使用するシステムでのセキュリティの検討　81
4.11.1　外部ネットワークを使用するシステムで想定されるセキュリティ上の脅威　81
4.11.2　ファイアウォールと侵入検知システムを配置する　82
4.11.3　SSL アクセラレータを使用して暗号通信を処理する　94
4.11.4　アプリケーションでユーザを認証する　95

第 3 編　機能解説

5　統合ユーザ管理による認証　98
5.1　この章の構成　99
5.2　統合ユーザ管理の概要　100
5.2.1　統合ユーザ管理の目的　100
5.2.2　レルムを使用したユーザ管理とユーザマッピング　100
5.2.3　Java 標準仕様（JAAS）に準拠したユーザ認証の概要　102
5.2.4　統合ユーザ管理で使用するユーザ情報の管理方法　105
5.2.5　ユーザ認証の有効期間と認証状態の引き継ぎ　109
5.2.6　統合ユーザ管理の処理の流れ　111
5.3　標準ログインモジュールによるユーザ認証の仕組み　113
5.3.1　標準ログインモジュールの種類と機能　113
5.3.2　WebPasswordLoginModule　115
5.3.3　WebCertificateLoginModule　116
5.3.4　WebPasswordLDAPLoginModule　117
5.3.5　WebPasswordJDBCLoginModule　119
5.3.6　DelegationLoginModule　121
5.3.7　WebSSOLoginModule　122
5.3.8　標準ログインモジュールによるリポジトリへのアクセス　124
5.3.9　パスワード認証時の暗号拡張　125
5.3.10　ログインモジュールが使用するコンフィグレーションファイルのパラメタ　126
5.4　統合ユーザ管理のセッション管理　130
5.4.1　セッションの種類　130
5.4.2　ログインしたユーザ ID の登録　130
5.4.3　統合ユーザ管理のセッションに登録したユーザ ID の削除　131
5.4.4　JAAS のコンフィグレーションファイルの定義例　132
5.5　シングルサインオンの利用方法　134
5.5.1　シングルサインオンをするために必要な手順　134
5.5.2　既存アプリケーションで行っているユーザ管理の適用　134
5.6　カスタムログインモジュールの利用　136
5.6.1　カスタムログインモジュールの概要　136
5.6.2　カスタムログインモジュールの呼び出し　137
5.7　ユーザ情報の管理　138
5.7.1　LDAP ディレクトリサーバへのユーザ情報の登録　138
5.7.2　LDAP ディレクトリサーバの多重化による接続フェイルオーバ　138
5.8　統合ユーザ管理フレームワークが提供する API　141
5.8.1　JSP タグライブラリ　141
5.8.2　統合ユーザ管理フレームワークのライブラリ　141

5.9 統合ユーザ管理フレームワークによるユーザ認証の実装 143
5.10 API を使用したユーザ認証の実装 144
5.10.1 ログインの実装（API を使用する場合） 144
5.10.2 ユーザ ID の取得の実装（API を使用する場合） 144
5.10.3 ユーザ属性の取得の実装（API を使用する場合） 145
5.10.4 認証に成功した Subject を HttpSession に登録する実装 146
5.10.5 ログアウトの実装（API を使用する場合） 147
5.10.6 ログイン状態のチェック（API を使用する場合） 147
5.10.7 パスワード認証時の暗号拡張の実装 147
5.10.8 API による実装時の注意事項 148
5.11 タグライブラリを使用したユーザ認証の実装 150
5.11.1 ログインの実装（タグライブラリを使用する場合） 150
5.11.2 ユーザ ID の取得の実装（タグライブラリを使用する場合） 152
5.11.3 ユーザ属性の取得の実装（タグライブラリを使用する場合） 152
5.11.4 ログアウトの実装（タグライブラリを使用する場合） 153
5.11.5 uatags.jar および uatags.tld のコピーと DD での定義 153
5.12 カスタムログインモジュールを使用したユーザ認証の実装 154
5.12.1 標準ログインモジュールと連携するための実装 154
5.12.2 カスタムログインモジュールの実装時の留意点 155
5.12.3 カスタムログインモジュールの実装例 156
5.13 統合ユーザ管理機能の設定手順 161
5.14 レルム名の決定 164
5.15 LDAP ディレクトリサーバの設定 165
5.15.1 LDAP ディレクトリサーバの設置 165
5.15.2 ユーザの登録とアクセス権の設定 165
5.15.3 オブジェクトクラスとユーザ定義属性の拡張 166
5.16 ユーザ情報の登録 168
5.16.1 コマンドによる登録 168
5.16.2 統合ユーザ管理フレームワークのライブラリによる登録 169
5.16.3 登録するユーザ情報の文法 169
5.16.4 Active Directory を使用する場合の設定 170
5.17 暗号鍵ファイルの作成（シングルサインオンを使用する場合） 174
5.17.1 暗号鍵ファイルの作成 174
5.17.2 暗号鍵ファイルの変更 174
5.18 ユーザ情報の登録（シングルサインオンを使用する場合） 175
5.18.1 コマンドによる登録 175
5.18.2 統合ユーザ管理フレームワークのライブラリによる登録 175
5.18.3 登録するユーザ情報の文法 176
5.19 コンフィグレーションファイルの作成 177

5.19.1 jaas.conf の作成 177
5.19.2 ua.conf の作成 179
5.19.3 コンフィグレーションファイルの設定例 180
5.20 JavaVM のプロパティの設定 187
5.21 ファイルのデプロイ 189

6 アプリケーションの設定による認証 190
6.1 この章の構成 191
6.2 DD の設定による Web コンテナのユーザ認証 192
6.2.1 DD の設定による Web コンテナのユーザ認証の機能 192
6.2.2 DD での定義 194
6.2.3 実行環境での設定（J2EE アプリケーションの設定） 194
6.2.4 認証機能を併用するときの注意 194
6.3 セキュリティアイデンティティを使用した認証 196
6.3.1 セキュリティアイデンティティの機能 196
6.3.2 EJB クライアントアプリケーションでのセキュリティの実装 197
6.3.3 セキュリティアイデンティティを使用した認証の設定 199

7 SSL/TLS 使用による認証情報とデータの暗号化 200
7.1 この章の構成 201
7.2 SSL 使用による認証情報とデータの暗号化 202
7.2.1 Web サーバの認証機能 202
7.2.2 HTTP Server の SSL の設定 203

8 API による直接接続を使用する負荷分散機の運用管理機能からの制御 204
8.1 この章の構成 205
8.2 API による直接接続を使用する負荷分散機 206
8.3 運用管理機能が実行する負荷分散機の API 207
8.3.1 Management Server（Smart Composer 機能）が実行する負荷分散機の API 207
8.3.2 仮想サーバマネージャが実行する負荷分散機の API 208
8.4 負荷分散機の接続環境の設定 210
8.4.1 アクセスリスト（ACL）の設定内容（ACOS の場合） 210
8.4.2 cookie パーシステンステンプレートの作成 210
8.4.3 トラストストアの設定 211
8.4.4 hosts ファイルの設定（BIG-IP の場合） 211
8.5 Management Server（Smart Composer 機能）での負荷分散機の接続情報の設定 212
8.6 仮想サーバマネージャでの負荷分散機の接続情報の設定 213
8.6.1 負荷分散機の接続情報を仮想サーバマネージャで定義するための設定 213
8.6.2 負荷分散機の接続情報を管理ユニットで定義するための設定 213

第 4 編　設定操作

9　サーバ管理コマンドによるセキュリティロールとアプリケーションの操作 215

9.1　この章の構成　216
9.2　セキュリティロールの設定　217
9.2.1　ユーザの設定　217
9.2.2　ロールの設定　217
9.3　セキュリティロールのリファレンス定義　221
9.3.1　Enterprise Bean のセキュリティロールリファレンスの定義　221
9.3.2　サーブレットと JSP のセキュリティロールリファレンスの定義　222
9.4　セキュリティの定義（メソッドパーミッション）　224
9.4.1　Enterprise Bean のメソッドパーミッション　224
9.5　セキュリティの定義（セキュリティアイデンティティ）　227
9.5.1　Enterprise Bean のセキュリティアイデンティティ　227
9.5.2　サーブレットと JSP のセキュリティアイデンティティ　229

10　運用管理ポータルによる統合ユーザ管理の運用　231

10.1　この章の構成　232
10.2　統合ユーザ管理のリソースの監視　233
10.2.1　統合ユーザ管理のリソース監視で確認できる項目　233
10.2.2　操作する画面（統合ユーザ管理のリソース監視）　234
10.3　リポジトリ管理によるユーザ情報の編集　235
10.3.1　ユーザ情報リポジトリに対するユーザ情報の編集　235
10.3.2　シングルサインオン情報リポジトリに対するシングルサインオン用のユーザ情報の編集　238
10.3.3　操作する画面（統合ユーザ管理のリポジトリ管理）　239
10.4　「リポジトリ管理」での注意事項　240

11　運用管理ポータルによるリポジトリ管理（統合ユーザ管理）　241

11.1　この章の構成　242
11.2　「リポジトリ管理」のツリーペインの構成　243
11.3　リポジトリ管理　244
11.3.1　バインド情報の設定　244
11.4　レルム管理　246
11.4.1　レルムの作成　246
11.4.2　暗号鍵ファイルの設定　248
11.4.3　ユーザエントリのスキーマ定義　249
11.4.4　ユーザエントリのスキーマ定義（シングルサインオン用）　252
11.4.5　ユーザエントリの作成　254
11.4.6　ユーザエントリの作成（シングルサインオン用）　257
11.4.7　ユーザエントリの検索　259

11.4.8　レルムの削除　261
11.4.9　ユーザエントリの編集　262
11.4.10　ユーザエントリの編集（シングルサインオン用）　264
11.4.11　ユーザエントリの削除　267

12　リソース監視（統合ユーザ管理）　269
12.1　この章の構成　270
12.2　「リソース監視」のツリーペインの構成　271
12.2.1　「リソース監視」のホストビューの構成　271
12.2.2　「リソース監視」のサーバビューの構成　271
12.3　ログインセッションの監視　273
12.3.1　ログインセッションモニタの表示　273
12.3.2　統合ユーザ管理のセッションの停止　275
12.4　LDAP 接続モニタの監視　276
12.4.1　LDAP 接続プールモニタの表示　276
12.4.2　LDAP 接続プールの空き待ち監視のリセット　278
12.4.3　LDAP 接続の定義情報の表示　279
12.4.4　LDAP 接続の障害情報の表示　283
12.5　JDBC 接続モニタの監視　285
12.5.1　JDBC 接続プールモニタの表示　285
12.5.2　JDBC 接続プールの空き待ち監視のリセット　287
12.5.3　JDBC 接続の定義情報の表示　288
12.5.4　JDBC 接続の障害情報の表示　291

第 5 編　リファレンス

13　統合ユーザ管理で使用するコマンド　293
13.1　統合ユーザ管理で使用するコマンドの一覧　294
13.2　統合ユーザ管理で使用するコマンドの詳細　295
convpw（パスワードの暗号化）　295
ssoexport（シングルサインオン情報リポジトリの参照）　296
ssogenkey（暗号鍵ファイルの作成）　298
ssoimport（シングルサインオン情報リポジトリの登録）　299
uachpw（パスワードの変更）　302

14　統合ユーザ管理で使用するファイル　304
14.1　統合ユーザ管理で使用するファイルの一覧　305
14.2　統合ユーザ管理で使用するファイルの詳細　306
14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）　306
14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）　313
14.3　シングルサインオン用認証情報の CSV 形式ファイル　322

14.3.1 CSV 形式ファイルの基本仕様 322
14.3.2 ユーザ情報を取得するための定義ファイル 322
14.3.3 ユーザ情報を追加および変更するための定義ファイル 323
14.3.4 ユーザマッピングと認証情報の定義ファイル 324
14.3.5 CSV 形式ファイルの記述例 325
14.3.6 ラインオペレーション 326

15 統合ユーザ管理フレームワークで使用する API 328
15.1 統合ユーザ管理フレームワークで使用する API の一覧 329
15.2 AttributeEntry クラス 331
AttributeEntry コンストラクタ 332
getAlias メソッド 332
getAttributeName メソッド 333
getSubcontext メソッド 333
setAlias メソッド 334
setAttributeName メソッド 334
setSubcontext メソッド 335
15.3 ChangeDataFailedException クラス 336
ChangeDataFailedException コンストラクタ 336
15.4 DelegationLoginModule クラス 337
15.5 LdapSSODataManager クラス 338
LdapSSODataManager コンストラクタ 339
addSSOData メソッド 339
addSSODataListener メソッド 340
getSSOData メソッド 341
getSSODataListeners メソッド 342
listUsers メソッド（形式 1） 342
listUsers メソッド（形式 2） 343
modifySSOData メソッド 344
removeSSOData メソッド 345
removeSSODataListener メソッド 346
15.6 LdapUserDataManager クラス 348
LdapUserDataManager コンストラクタ 349
addUserData メソッド（形式 1） 351
addUserData メソッド（形式 2） 352
getUserData メソッド 354
listUsers メソッド（形式 1） 355
listUsers メソッド（形式 2） 355
modifyUserData メソッド 356
removeUserData メソッド 357
15.7 LdapUserEnumeration インタフェース 359
close メソッド 359
hasMore メソッド 360

hasMoreElements メソッド　361
next メソッド　361
nextElement メソッド　362
15.8　LoginUtil クラス　363
check メソッド（形式 1）　363
check メソッド（形式 2）　364
15.9　ObjectClassEntry クラス　366
ObjectClassEntry コンストラクタ　366
getObjectClasses メソッド　367
getSubcontext メソッド　368
setObjectClasses メソッド　368
setSubcontext メソッド　369
15.10　PasswordCryptography インタフェース　370
encrypt メソッド　370
15.11　PasswordUtil クラス　371
changePassword メソッド　371
15.12　Principal インタフェース　373
15.13　SSOData クラス　374
SSOData コンストラクタ　374
getMapping メソッド　375
getMappingRealms メソッド　375
getPublicData メソッド　376
removeMapping メソッド　376
setMapping メソッド　377
setPublicData メソッド　378
setSecretData メソッド　378
15.14　SSODataEvent クラス　380
SSODataEvent コンストラクタ　380
getOldPublicData メソッド　381
getOldSecretData メソッド　382
getPublicData メソッド　382
getSecretData メソッド　383
getUserId メソッド　383
15.15　SSODataListener インタフェース　384
ssoDataAdded メソッド　385
ssoDataModified メソッド　386
ssoDataRemoved メソッド　386
15.16　SSODataListenerException クラス　388
SSODataListenerException コンストラクタ　388
getException メソッド　389
getListeners メソッド　389
setException メソッド　390
15.17　UserAttributes インタフェース　391
addAttribute メソッド　392

getAttribute メソッド　392
getAttributeNames メソッド　393
getAttributes メソッド　394
removeAttribute メソッド　395
size メソッド　395
15.18 UserData クラス　397
UserData コンストラクタ　397
addAttribute メソッド　398
getAttribute メソッド　399
getAttributeNames メソッド　399
getAttributes メソッド　400
removeAttribute メソッド　400
setPassword メソッド　401
size メソッド　401
15.19 WebCertificateCallback クラス　403
WebCertificateCallback コンストラクタ　404
getAttributeEntries メソッド　404
getRequest メソッド　405
getResponse メソッド　405
getSubjectID メソッド　406
getTagEntry メソッド　406
getTagID メソッド　407
setAttributeEntries メソッド　407
setRequest メソッド　408
setResponse メソッド　408
setSubjectID メソッド　409
setTagEntry メソッド　410
setTagID メソッド　410
15.20 WebCertificateHandler クラス　411
WebCertificateHandler コンストラクタ　411
handle メソッド　413
15.21 WebCertificateLoginModule クラス　414
15.22 WebLogoutCallback クラス　415
WebLogoutCallback コンストラクタ　415
getSession メソッド　416
getUserID メソッド　416
setSession メソッド　417
setUserID メソッド　417
15.23 WebLogoutHandler クラス　419
WebLogoutHandler コンストラクタ　419
handle メソッド　420
15.24 WebPasswordCallback クラス　421
WebPasswordCallback コンストラクタ　422
getAttributeEntries メソッド　423

getName メソッド　423
getOption メソッド　424
getPassword メソッド　424
getRequest メソッド　425
getResponse メソッド　425
getTagEntry メソッド　426
getTagID メソッド　426
setAttributeEntries メソッド　427
setName メソッド　427
setOption メソッド　428
setPassword メソッド　429
setRequest メソッド　429
setResponse メソッド　430
setTagEntry メソッド　430
setTagID メソッド　431
15.25　WebPasswordHandler クラス　432
WebPasswordHandler コンストラクタ　432
handle メソッド　434
15.26　WebPasswordJDBCLoginModule クラス　436
15.27　WebPasswordLDAPLoginModule クラス　437
15.28　WebPasswordLoginModule クラス　438
15.29　WebSSOCallback クラス　439
WebSSOCallback コンストラクタ　440
getRequest メソッド　440
getResponse メソッド　441
getTagEntry メソッド　441
getTagID メソッド　442
setRequest メソッド　442
setResponse メソッド　443
setTagEntry メソッド　443
setTagID メソッド　444
15.30　WebSSOHandler クラス　445
WebSSOHandler コンストラクタ　445
handle メソッド　446
15.31　WebSSOLoginModule クラス　448
15.32　例外クラス　449
15.32.1　JAAS のログインモジュールの例外クラス　449
15.32.2　この製品で提供する API の例外クラス　451

16　統合ユーザ管理フレームワークで使用するタグライブラリ　453
16.1　タグライブラリのタグの一覧　454
16.2　タグライブラリのタグの詳細　455
16.2.1　<ua:attributeEntries>Entries</ua:attributeEntries>タグ　455

16.2.2 <ua:attributeEntry/>タグ 456
16.2.3 <ua:chpw/>タグ 456
16.2.4 <ua:exception>Body</ua:exception>タグ 458
16.2.5 <ua:getPrincipalName/>タグ 458
16.2.6 <ua:getAttribute/>タグ 459
16.2.7 <ua:getAttributes/>タグ 460
16.2.8 <ua:getAttributeNames/>タグ 461
16.2.9 <ua:login/>タグ 462
16.2.10 <ua:logout/>タグ 464
16.2.11 <ua:notLogin>Body</ua:notLogin>タグ 464

17 EJB クライアントアプリケーションの実装で使用する API 466

17.1 LoginInfoManager クラス 467
getLoginInfoManager メソッド 468
login メソッド 468
logout メソッド 469

18 API による直接接続を使用する負荷分散機の制御で使用するファイル 471

18.1 API による直接接続を使用する負荷分散機の制御で使用するファイルの一覧 472
18.2 API による直接接続を使用する負荷分散機の制御で使用するファイルの詳細 473
18.2.1 lb.properties（負荷分散機定義プロパティファイル） 473
18.2.2 <LB 接続情報の識別名>.properties（仮想サーバマネージャ側の負荷分散機接続設定プロパティファイル） 475
18.2.3 tierlb.properties（ティア側の負荷分散機接続設定プロパティファイル） 478

19 セキュリティ管理機能で出力されるメッセージ 481

19.1 メッセージの記述形式 482
19.2 KDCGF で始まるメッセージ 484
19.3 KDCGK で始まるメッセージ 492
19.4 KDCGS で始まるメッセージ 495
19.5 KDCGW で始まるメッセージ 498
19.6 KEOS02000 から KEOS09999 までのメッセージ 500
19.7 KEXS で始まるメッセージ 505
19.8 SSL についてのメッセージ 515
19.8.1 メッセージの記述形式 515
19.8.2 注意事項 516
19.8.3 AH で始まるメッセージ 516
19.8.4 KH で始まるメッセージ 528

付録 532

付録 A 各バージョンでの主な機能変更 533

付録 A.1　11-10 での主な機能変更　533
付録 A.2　11-00 での主な機能変更　534
付録 A.3　09-87 での主な機能変更　536
付録 A.4　09-80 での主な機能変更　536
付録 A.5　09-70 での主な機能変更　537
付録 A.6　09-60 での主な機能変更　540
付録 A.7　09-50 での主な機能変更　540
付録 A.8　09-00 での主な機能変更　545
付録 A.9　08-70 での主な機能変更　548
付録 A.10　08-53 での主な機能変更　551
付録 A.11　08-50 での主な機能変更　552
付録 A.12　08-00 での主な機能変更　555
付録 B　例外リストの登録（Windows の場合）　559
付録 C　用語解説　564

索引　565

1 アプリケーションサーバの機能

この章では，アプリケーションサーバの機能の分類と目的，および機能とマニュアルの対応について説明します。また，このバージョンで変更した機能についても説明しています。

1.1　機能の分類

アプリケーションサーバは，Java EE 8 に対応した J2EE サーバを中心としたアプリケーションの実行環境を構築したり，実行環境上で動作するアプリケーションを開発したりするための製品です。Java EE の標準仕様に準拠した機能や，アプリケーションサーバで独自に拡張された機能など，多様な機能を使用できます。目的や用途に応じた機能を選択して使用することで，信頼性の高いシステムや，処理性能に優れたシステムを構築・運用できます。

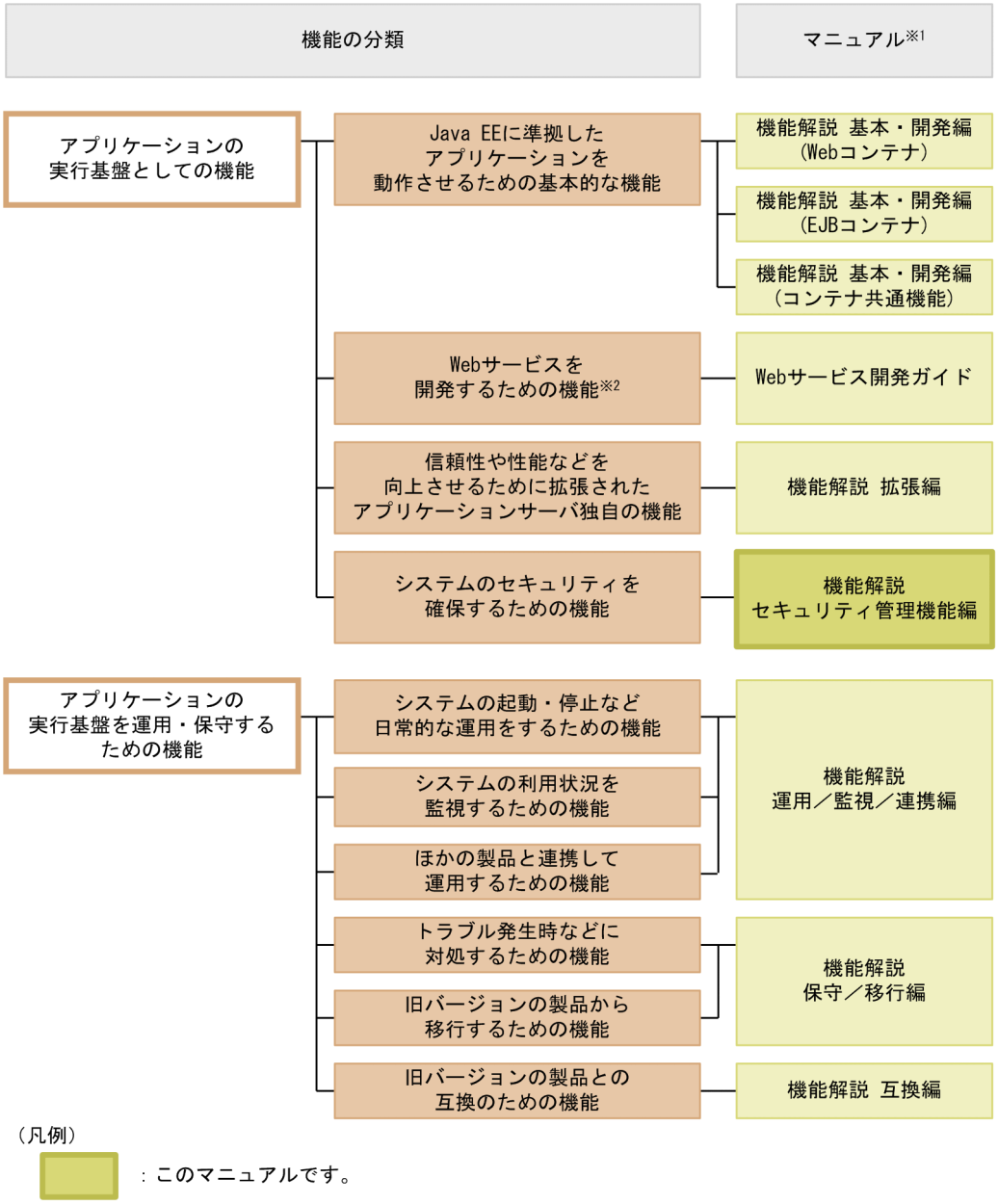
アプリケーションサーバの機能は，大きく分けて，次の二つに分類できます。

- アプリケーションの実行基盤としての機能
- アプリケーションの実行基盤を運用・保守するための機能

二つの分類は，機能の位置づけや用途によって，さらに詳細に分類できます。アプリケーションサーバのマニュアルは，機能の分類に合わせて提供しています。

アプリケーションサーバの機能の分類と対応するマニュアルについて，次の図に示します。

図 1-1　アプリケーションサーバの機能の分類と対応するマニュアル



注※1

マニュアル名称の「アプリケーションサーバ」を省略しています。

注※2

アプリケーションサーバでは，SOAP Web サービスと RESTful Web サービスを実行できます。目的によっては，マニュアル「アプリケーションサーバ Web サービス開発ガイド」以外の次のマニュアルも参照してください。

SOAP アプリケーションを開発・実行する場合

- アプリケーションサーバ SOAP アプリケーション開発の手引

SOAP Web サービスや SOAP アプリケーションのセキュリティを確保する場合

- XML Security - Core ユーザーズガイド

- アプリケーションサーバ Web サービスセキュリティ構築ガイド

XML の処理について詳細を知りたい場合

- XML Processor ユーザーズガイド

ここでは，機能の分類について，マニュアルとの対応と併せて説明します。

1.1.1　アプリケーションの実行基盤としての機能

アプリケーションとして実装されたオンライン業務やバッチ業務を実行する基盤となる機能です。システムの用途や求められる要件に応じて，使用する機能を選択します。

アプリケーションの実行基盤としての機能を使用するかどうかは，システム構築やアプリケーション開発よりも前に検討する必要があります。

アプリケーションの実行基盤としての機能について，分類ごとに説明します。

(1)　アプリケーションを動作させるための基本的な機能（基本・開発機能）

アプリケーション（J2EE アプリケーション）を動作させるための基本的な機能が該当します。主に J2EE サーバの機能が該当します。

アプリケーションサーバでは，Java EE 8 に対応した J2EE サーバを提供しています。J2EE サーバでは，標準仕様に準拠した機能のほか，アプリケーションサーバ独自の機能も提供しています。

基本・開発機能は，機能を使用する J2EE アプリケーションの形態に応じて，さらに三つに分類できます。アプリケーションサーバの機能解説のマニュアルは，この分類に応じて分冊されています。

それぞれの分類の概要を説明します。

- **Web アプリケーションを実行するための機能（Web コンテナ）**
  Web アプリケーションの実行基盤である Web コンテナの機能，および Web コンテナと Web サーバが連携して実現する機能が該当します。
- **Enterprise Bean を実行するための機能（EJB コンテナ）**
  Enterprise Bean の実行基盤である EJB コンテナの機能が該当します。また，Enterprise Bean を呼び出す EJB クライアントの機能も該当します。
- **Web アプリケーションと Enterprise Bean の両方で使用する機能（コンテナ共通機能）**
  Web コンテナ上で動作する Web アプリケーションおよび EJB コンテナ上で動作する Enterprise Bean の両方で使用できる機能が該当します。

(2)　Web サービスを開発するための機能

Web サービスの実行環境および開発環境としての機能が該当します。

アプリケーションサーバでは，次のエンジンを提供しています。

- JAX-WS 仕様に従った SOAP メッセージのバインディングを実現する JAX-WS エンジン
- JAX-RS 仕様に従った RESTful HTTP メッセージのバインディングを実現する JAX-RS エンジン

(3) 信頼性や性能などを向上させるために拡張されたアプリケーションサーバ独自の機能（拡張機能）

アプリケーションサーバで独自に拡張された機能が該当します。バッチサーバ，CTM，データベースなど，J2EE サーバ以外のプロセスを使用して実現する機能も含まれます。

アプリケーションサーバでは，システムの信頼性を高め，安定稼働を実現するための多様な機能が拡張されています。また，J2EE アプリケーション以外のアプリケーション（バッチアプリケーション）を Java の環境で動作させる機能も拡張しています。

(4) システムのセキュリティを確保するための機能（セキュリティ管理機能）

アプリケーションサーバを中心としたシステムのセキュリティを確保するための機能が該当します。不正なユーザからのアクセスを防止するための認証機能や，通信路での情報漏えいを防止するための暗号化機能などがあります。

1.1.2 アプリケーションの実行基盤を運用・保守するための機能

アプリケーションの実行基盤を効率良く運用したり，保守したりするための機能です。システムの運用開始後に，必要に応じて使用します。ただし，機能によっては，あらかじめ設定やアプリケーションの実装が必要なものがあります。

アプリケーションの実行基盤を運用・保守するための機能について，分類ごとに説明します。

(1) システムの起動・停止など日常的な運用をするための機能（運用機能）

システムの起動や停止，アプリケーションの開始や停止，入れ替えなどの，日常運用で使用する機能が該当します。

(2) システムの利用状況を監視するための機能（監視機能）

システムの稼働状態や，リソースの枯渇状態などを監視するための機能が該当します。また，システムの操作履歴など，監査で使用する情報を出力する機能も該当します。

(3) ほかの製品と連携して運用するための機能（連携機能）

JP1 やクラスタソフトウェアなど，ほかの製品と連携して実現する機能が該当します。

(4)　トラブル発生時などに対処するための機能（保守機能）

トラブルシューティングのための機能が該当します。トラブルシューティング時に参照する情報を出力するための機能も含みます。

(5)　旧バージョンの製品から移行するための機能（移行機能）

旧バージョンのアプリケーションサーバから新しいバージョンのアプリケーションサーバに移行するための機能が該当します。

(6)　旧バージョンの製品との互換のための機能（互換機能）

旧バージョンのアプリケーションサーバとの互換用の機能が該当します。なお，互換機能については，対応する推奨機能に移行することをお勧めします。

1.1.3　機能とマニュアルの対応

アプリケーションサーバの機能解説のマニュアルは，機能の分類に合わせて分冊されています。

機能の分類と，それぞれの機能について説明しているマニュアルとの対応を次の表に示します。

表 1-1　機能の分類と機能解説のマニュアルの対応

分類	機能	マニュアル※1
基本・開発機能	Web コンテナ	基本・開発編(Web コンテナ)
	JSF および JSTL の利用	
	JAX-RS 2.1 の利用	
	WebSocket	
	NIO HTTP サーバ	
	サーブレットおよび JSP の実装	
	セッションマネージャの指定機能	
	EJB コンテナ	基本・開発編(EJB コンテナ)
	EJB クライアント	
	Enterprise Bean 実装時の注意事項	
	ネーミング管理	基本・開発編(コンテナ共通機能)
	リソース接続とトランザクション管理	
	OpenTP1 からのアプリケーションサーバの呼び出し（TP1 インバウンド連携機能）	

分類	機能	マニュアル※1
	JPA 2.2 の利用	
	CJMS プロバイダ	
	JavaMail の利用	
	アプリケーションサーバでの CDI の利用	
	アプリケーションサーバでの Bean Validation の利用	
	Java Batch	
	JSON-P	
	JSON-B	
	Concurrency Utilities	
	アプリケーションの属性管理	
	アノテーションの使用	
	J2EE アプリケーションの形式とデプロイ	
	コンテナ拡張ライブラリ	
	ライブラリ競合回避機能	
拡張機能	バッチサーバによるアプリケーションの実行	拡張編
	CTM によるリクエストのスケジューリングと負荷分散	
	バッチアプリケーションのスケジューリング	
	J2EE サーバ間のセッション情報の引き継ぎ（セッションフェイルオーバ機能）	
	データベースセッションフェイルオーバ機能	
	明示管理ヒープ機能を使用した FullGC の抑止	
	アプリケーションのユーザログ出力	
セキュリティ管理機能	統合ユーザ管理による認証	セキュリティ管理機能編※2
	アプリケーションの設定による認証	
	SSL/TLS 通信での TLSv1.2 の使用	
	API による直接接続を使用する負荷分散機の運用管理機能からの制御	
運用機能	システムの起動と停止	運用／監視／連携編
	J2EE アプリケーションの運用	
監視機能	稼働情報の監視（稼働情報収集機能）	
	リソースの枯渇監視	

分類	機能	マニュアル※1
	監査ログ出力機能	
	データベース監査証跡連携機能	
	運用管理コマンドによる稼働情報の出力	
	Management イベントの通知と Management アクションによる処理の自動実行	
	CTM の稼働統計情報の収集	
	コンソールログの出力	
連携機能	JP1 と連携したシステムの運用	
	システムの集中監視（JP1/IM との連携）	
	ジョブによるシステムの自動運転（JP1/AJS との連携）	
	監査ログの収集および一元管理（JP1/Audit Management - Manager との連携）	
	クラスタソフトウェアとの連携	
	1:1 系切り替えシステム（クラスタソフトウェアとの連携）	
	相互系切り替えシステム（クラスタソフトウェアとの連携）	
	N:1 リカバリシステム（クラスタソフトウェアとの連携）	
	ホスト単位管理モデルを対象にした系切り替えシステム（クラスタソフトウェアとの連携）	
保守機能	トラブルシューティング関連機能	保守／移行編
	性能解析トレースを使用した性能解析	
	製品の JavaVM（以降，JavaVM と略す場合があります）の機能	
移行機能	旧バージョンのアプリケーションサーバからの移行	
	推奨機能への移行	
互換機能	基本・開発機能の互換機能	互換編
	拡張機能の互換機能	

注※1　マニュアル名称の「アプリケーションサーバ 機能解説」を省略しています。

注※2　このマニュアルです。

1.2　システムの目的と機能の対応

アプリケーションサーバでは，構築・運用するシステムの目的に合わせて，適用する機能を選択する必要があります。

この節では，セキュリティを確保するためにアプリケーションサーバが提供しているそれぞれの機能をどのようなシステムの場合に使用するとよいかを示します。機能ごとに，次の項目への対応を示しています。

- **信頼性**
  高い信頼が求められるシステムの場合に使用するとよい機能です。
  アベイラビリティ（安定稼働性）およびフォールトトレランス（耐障害性）を高める機能や，ユーザ認証などのセキュリティを高めるための機能が該当します。
- **性能**
  性能を重視したシステムの場合に使用するとよい機能です。
  システムのパフォーマンスチューニングで使用する機能などが該当します。
- **運用・保守**
  効率の良い運用・保守をしたい場合に使用するとよい機能です。
- **拡張性**
  システム規模の拡大・縮小および構成の変更への柔軟な対応が必要な場合に使用するとよい機能です。
- **そのほか**
  そのほかの個別の目的に対応するための機能です。

また，セキュリティを確保するためにアプリケーションサーバが提供している機能には，Java EE 標準機能とアプリケーションサーバが独自に拡張した機能があります。機能を選択するときには，必要に応じて，Java EE 標準への準拠についても確認してください。

1.2.1　認証のための機能

認証のための機能を次の表に示します。システムの目的に合った機能を選択してください。機能の詳細については，参照先を確認してください。

表 1-2　認証のための機能とシステムの目的の対応

機能	システムの目的					Java EE 標準への準拠		参照先
	信頼性	性能	運用・保守	拡張性	そのほか	標準	拡張	
統合ユーザ管理	○	－	○	－	－	○	○	5 章
アプリケーションの設定による認証	○	－	－	－	－	○	○	6 章

（凡例）　○：対応する　　－：対応しない

注　「Java EE 標準への準拠」の「標準」と「拡張」の両方に○が付いている機能は，Java EE 標準の機能にアプリケーションサーバ独自の機能が拡張されていることを示します。「拡張」だけに○が付いている機能はアプリケーションサーバ独自の機能であることを示します。

1.2.2　暗号化のための機能

暗号化のための機能を次の表に示します。システムの目的に合った機能を選択してください。機能の詳細については，参照先を確認してください。

表 1-3　暗号化のための機能とシステムの目的の対応

機能	システムの目的					Java EE 標準への準拠		参照先
	信頼性	性能	運用・保守	拡張性	そのほか	標準	拡張	
SSL/TLS 使用による認証情報とデータの暗号化	○	－	－	－	－	○	○	7 章，マニュアル「HTTP Server」
Web サービスセキュリティの機能による SOAP メッセージの暗号化	○	－	－	－	－	○	○	マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」
Web サービスセキュリティの機能で使用する XML による SOAP メッセージの暗号化での XML 署名・暗号処理機能	○	－	－	－	－	○	○	マニュアル「XML Security - Core ユーザーズガイド」

（凡例）　○：対応する　　－：対応しない

注　「Java EE 標準への準拠」の「標準」と「拡張」の両方に○が付いている機能は，Java EE 標準の機能にアプリケーションサーバ独自の機能が拡張されていることを示します。「拡張」だけに○が付いている機能はアプリケーションサーバ独自の機能であることを示します。

1.2.3　不正処理の実行防止のための機能

不正処理の実行防止のための機能を次の表に示します。システムの目的に合った機能を選択してください。機能の詳細については，参照先を確認してください。

表 1-4　不正処理の実行防止のための機能とシステムの目的の対応

機能	システムの目的					Java EE 標準への準拠		参照先
	信頼性	性能	運用・保守	拡張性	そのほか	標準	拡張	
SecurityManager による Web コンテナの実行時の保護	○	－	－	－	－	○	○	2.2.5

（凡例）　○：対応する　　－：対応しない

注　「Java EE 標準への準拠」の「標準」と「拡張」の両方に○が付いている機能は，Java EE 標準の機能にアプリケーションサーバ独自の機能が拡張されていることを示します。「拡張」だけに○が付いている機能はアプリケーションサーバ独自の機能であることを示します。

1.2.4　そのほかの機能

ほかのプログラムとの連携によるセキュアな通信の要求に従う機能を次の表に示します。システムの目的に合った機能を選択してください。機能の詳細については，参照先を確認してください。

表 1-5　ほかのプログラムとの連携によるセキュアな通信の要求に従う機能とシステムの目的の対応

機能	システムの目的					Java EE 標準への準拠		参照先
	信頼性	性能	運用・保守	拡張性	そのほか	標準	拡張	
API による直接接続を使用する負荷分散機の運用管理機能からの制御	－	－	－	－	○	－	－	8 章

（凡例）　○：対応する　　－：対応しない

1.3　このマニュアルに記載している機能の説明

ここでは，このマニュアルで機能を説明するときの分類の意味と，分類を示す表の例について説明します。

1.3.1　分類の意味

このマニュアルでは，各機能について，次の五つに分類して説明しています。マニュアルを参照する目的によって，必要な個所を選択して読むことができます。

- 解説
  機能の解説です。機能の目的，特長，仕組みなどについて説明しています。機能の概要について知りたい場合にお読みください。
- 実装
  コーディングの方法や DD の記載方法などについて説明しています。アプリケーションを開発する場合にお読みください。
- 設定
  システム構築時に必要となるプロパティなどの設定方法について説明しています。システムを構築する場合にお読みください。
- 運用
  運用方法の説明です。運用時の手順や使用するコマンドの実行例などについて説明しています。システムを運用する場合にお読みください。
- 注意事項
  機能を使用するときの全般的な注意事項について説明しています。注意事項の説明は必ずお読みください。

1.3.2　分類を示す表の例

機能説明の分類については，表で説明しています。表のタイトルは，「この章の構成」または「この節の構成」となっています。

次に，機能説明の分類を示す表の例を示します。

機能説明の分類を示す表の例

表 X-1　この章の構成（○○機能）

分類	タイトル	参照先
解説	○○機能とは	X.1
実装	アプリケーションの実装	X.2

分類	タイトル	参照先
	DD および cosminexus.xml※での定義	X.3
設定	実行環境での設定	X.4
運用	○○機能を使用した運用	X.5
注意事項	○○機能使用時の注意事項	X.6

注※

cosminexus.xml については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)」の「16. アプリケーションの属性管理」を参照してください。

ポイント

cosminexus.xml を含まないアプリケーションのプロパティ設定

cosminexus.xml を含まないアプリケーションでは，実行環境へのインポート後にプロパティを設定，または変更します。設定済みのプロパティも実行環境で変更できます。

実行環境でのアプリケーションの設定は，サーバ管理コマンドおよび属性ファイルで実施します。サーバ管理コマンドおよび属性ファイルでのアプリケーションの設定については，マニュアル「アプリケーションサーバ アプリケーション設定操作ガイド」の「3.5.2　J2EE アプリケーションのプロパティの設定手順」を参照してください。

属性ファイルで指定するタグは，DD または cosminexus.xml と対応しています。DD または cosminexus.xml と属性ファイルのタグの対応についてはマニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「2.　アプリケーション属性ファイル（cosminexus.xml）」を参照してください。

なお，各属性ファイルで設定するプロパティは，アプリケーション統合属性ファイルでも設定できます。

1.4　アプリケーションサーバ 11-20 での主な機能変更

この節では，アプリケーションサーバ 11-20 での主な機能の変更について，変更目的ごとに説明します。

説明内容は次のとおりです。

- アプリケーションサーバ 11-20 で変更になった主な機能と，その概要を説明しています。機能の詳細については参照先の記述を確認してください。「参照先マニュアル」および「参照個所」には，その機能についての主な記載個所を記載しています。
- 「参照先マニュアル」に示したマニュアル名の「アプリケーションサーバ」は省略しています。

1.4.1　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 1-6　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
JSF 2.3 への対応	JSF 2.3 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	3 章
JAX-RS 2.1 への対応	JAX-RS 2.1 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	4 章
WebSocket 1.1 への対応	WebSocket 1.1 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	5 章
Servlet 4.0 への対応	Servlet 4.0 に対応しました。これに伴い，NIO HTTP サーバで HTTP/2 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	8 章
CDI Managed Bean での JPA 利用	CDI Managed Bean での JPA 利用に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	5 章
JPA 2.2 への対応	JPA 2.2 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	6 章
CDI 2.0 への対応	CDI 2.0 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	9 章
WEB-INF/lib 内の CDI への対応	WAR ファイルの WEB-INF/lib 以下の JAR ファイルから CDI を利用できるようにしました。	機能解説 基本・開発編 (コンテナ共通機能)	9 章
BV 2.0 への対応	BV 2.0 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	10 章
JSON-P 1.1 への対応	JSON-P 1.1 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	12 章
JSON-B 1.0 への対応	JSON-B 1.0 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	13 章

2 アプリケーションサーバでのセキュリティ管理

この章では，アプリケーションサーバのセキュリティを管理するための機能および方法について説明します。この章の内容を基に，確保したいセキュリティの内容やレベルに応じて，どの機能または方法を適用すればよいかを判断してください。

2.1　この章の構成

アプリケーションサーバでは，システムのセキュリティを管理するための機能を提供しています。これらの機能を目的に応じて適切に利用することで，必要なセキュリティを確保したシステムを構築・運用できます。

この章の構成を次の表に示します。

表 2–1　この章の構成（アプリケーションサーバでのセキュリティ管理）

分類	タイトル	参照先
解説	セキュリティを確保するための観点	2.2
	セキュリティを確保する方法および機能の詳細	2.3
注意事項	セキュリティを確保する方法および機能を使用する場合の注意事項	2.4

注　「実装」,「設定」，および「運用」について，この章での説明はありません。

2.2　セキュリティを確保するための観点

アプリケーションサーバのセキュリティを確保するためには，次の観点での検討が必要です。

- セキュリティを確保するためのシステム構成を検討する
- システムをセキュアに運用する方法を検討する
- 不正なユーザからのアクセスを防止する
- 通信路でのセキュリティを確保する
- 不正な処理の実行を防止する
- そのほかの観点を確認する

2.2.1　セキュリティを確保するためのシステム構成を検討する

セキュリティを確保するための機器またはソフトウェアをシステム上に適切に配置し，外部からの不正な侵入を防止するためのシステム構成を検討します。

ファイアウォールを使用すると，外部のネットワークと内部のネットワーク間のアクセスを制御できます。アクセスを許可するクライアントや通信をあらかじめ特定し，決められたルールに従って通信を許可したり破棄したりすることで，外部ネットワークからの不正なアクセスを防止します。さらに，**侵入検知システム**（IDS）を使用すると，通信回線を監視して，通信パターンによって不正なアクセスを判断し，対処できるようになります。

リバースプロキシサーバを配置すると，Web サーバ上に配置した重要な情報を含むコンテンツが悪意のあるクライアントから直接アクセスされることを防止できます。クライアントからのリクエストはリバースプロキシサーバが受け付け，コンテンツが配置された Web サーバへのアクセスはリバースプロキシサーバから実行するように制御できます。

また，暗号化によって通信路のセキュリティを確保している場合は，暗号化・複合化の処理を **SSL アクセラレータ**で実施することで，Web サーバやアプリケーションサーバに負荷を掛けることなく実現できます。

2.2.2　システムをセキュアに運用する

構築したシステムは，適切に運用することでセキュリティを確保できます。

運用の前に，どのユーザによる運用を許可するのか，どのような情報を管理するのか，システムの物理的な配置はどうするかなどを検討します。

また，検討した内容に沿って正しく運用されるためには，必要に応じて手順書を整備した上で，正しく運用されているかを適切にチェックする仕組みを作ることも必要です。

2.2.3　不正なユーザからのアクセスを防止する（認証機能）

システムで管理している情報を安全に管理し，セキュリティを確保するためには，**認証機能**によって不正なユーザからのシステムへのアクセスを防止することが有効です。

アプリケーションサーバでは，次の認証機能を提供しています。

- **統合ユーザ管理フレームワークを使用した，システムにログインするユーザの認証**

  システムにログインするユーザの情報を統合管理することで，複数のアプリケーションへのシングルサインオンを実現できる機能です。

- **<security-constraint>要素の設定による Web コンテナでの認証**

  Web コンテナの機能を使用して，許可したユーザだけにアプリケーションに対するアクセスを許可する機能です。認証で使用する情報は，DD（web.xml）の<security-constraint>要素で定義します。

- **<security-identity>要素の設定による EJB コンテナでの認証**

  EJB コンテナの機能を使用して，許可したユーザだけにアプリケーションに対するアクセスを許可する機能です。認証で使用する情報は，DD（ejb-jar.xml）または cosminexus.xml の<security-identity>要素で定義します。

  また，<method-permission>要素に設定したメソッドごとに，ロールに設定したユーザごとに実行できるメソッドを制御することもできます。

  なお，アプリケーションサーバが提供する API を使用して EJB クライアントアプリケーションを実装することで，EJB クライアントアプリケーションからのアクセスを認証することもできます。

2.2.4　通信路でのセキュリティを確保する（暗号化機能）

クライアントとアプリケーションサーバ間での通信路での情報の不正漏えいを防止するためには，やり取りする情報を暗号化することが有効です。

アプリケーションサーバでは，次の暗号化機能を提供しています。

- **SSL による暗号化**

  Web サーバおよび J2EE サーバの機能を使用して，通信路のデータを暗号化する機能です。

- **Web サービスセキュリティの機能による SOAP メッセージの暗号化**

  Web サービスセキュリティの機能を使用して，Web サービスで送受信する SOAP メッセージを暗号化したり，SOAP メッセージに XML 署名を付与したりできます。

2.2.5　不正な処理の実行を防止する

J2EE サーバで不正な処理が実行されることを防ぐために，Java SE の **SecurityManager** の機能による J2EE サーバの実行時の保護ができます。

J2EE サーバの実行時の保護では，次に示すような現象を防ぐことができます。

- 内部で System.exit()などを発行するような不正なサーブレット，EJB によって，J2EE サーバ全体が勝手に終了する。
- システムプロパティなどを勝手に書き換えるような不正なサーブレット，EJB によって，J2EE サーバの実行に不具合が生じる。

なお，J2EE サーバの保護はデフォルトで有効になっています。使用しない場合は，J2EE サーバを開始する cjstartsv コマンドに-nosecurity オプションを指定してください。

J2EE サーバの実行時保護を無効にした場合，java.lang.System の setSecurityManager メソッドは使用できません。使用した場合，J2EE サーバの実行に不具合が生じるおそれがあります。また，該当する J2EE サーバプロセスから EJB を呼び出す場合にダイナミッククラスローディングは使用できません。

2.2.6　そのほかの観点を確認する

ほかのプログラムとの連携でセキュアな通信を要求される場合，要求に従った設定をします。

2.3　セキュリティを確保する方法および機能の詳細

この章で説明した観点でセキュリティを確保する方法および機能の詳細は，次の表に示す参照先で説明しています。

表 2-2　セキュリティを確保する方法および機能の詳細

観点	機能	参照先
セキュリティを確保するためのシステム構成を検討する	－	3 章，4 章
システムをセキュアに運用する	－	4 章
不正なユーザからのアクセスを防止する（認証機能）	統合ユーザ管理フレームワークを使用した，システムにログインするユーザの認証	5 章
	DD の設定による Web コンテナでの認証	6.2
	セキュリティアイデンティティを使用した認証	6.3
通信路でのセキュリティを確保する（暗号化機能）	SSL による暗号化（Web サーバ側）	マニュアル「HTTP Server」
	SSL/TLS 使用による認証情報とデータの暗号化	7 章
	Web サービスセキュリティの機能による SOAP メッセージの暗号化	マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」
	Web サービスセキュリティの機能で使用する XML による SOAP メッセージの暗号化での XML 署名・暗号処理機能	マニュアル「XML Security - Core ユーザーズガイド」
不正な処理の実行を防止する	SecurityManager による Web コンテナの実行時の保護	2.2.5
そのほかの観点を確認する	API による直接接続を使用する負荷分散機の運用管理機能からの制御	8 章

（凡例）－：該当なし。

2.4　セキュリティを確保する方法および機能を使用する場合の注意事項

2.4.1　証明書について

アプリケーションサーバに含まれる cacerts 証明書ファイルには，証明書を同梱していません。証明書が必要な場合は，個別に入手してインポートする必要があります。インポート手順については，次の URL を参照してください。

Windows の場合

http://docs.oracle.com/javase/jp/8/docs/technotes/tools/windows/keytool.html

UNIX の場合

http://docs.oracle.com/javase/jp/8/docs/technotes/tools/unix/keytool.html

3 セキュリティを確保するためのシステム構成

この章では，J2EE アプリケーション実行基盤でセキュリティを確保するためのシステム構成について説明します。コンポーネントの種類に応じたファイアウォールの配置を検討する場合，および DMZ へのリバースプロキシの配置を検討する場合の構成について，システム構成例に従って説明します。

3.1　この章の構成

この章では，セキュリティを確保するためのシステム構成について説明します。この章の構成を次の表に示します。

表 3–1　この章の構成（セキュリティを確保するためのシステム構成）

分類	タイトル	参照先
解説	ファイアウォールを使用する構成を検討する	3.2
	DMZ へのリバースプロキシの配置を検討する	3.3

注　「実装」,「設定」,「運用」，および「注意事項」について，この章での説明はありません。

3.2　ファイアウォールを使用する構成を検討する

この節では，ファイアウォールを使用して，セキュリティを確保するための構成について説明します。

なお，ここでは，アクセスポイントになるコンポーネントの種類に応じたファイアウォールの配置位置について説明します。このほかのセキュリティの考え方については，「4.　セキュアなシステムの検討」を参照してください。

3.2.1　サーブレットと JSP に対するファイアウォールの配置

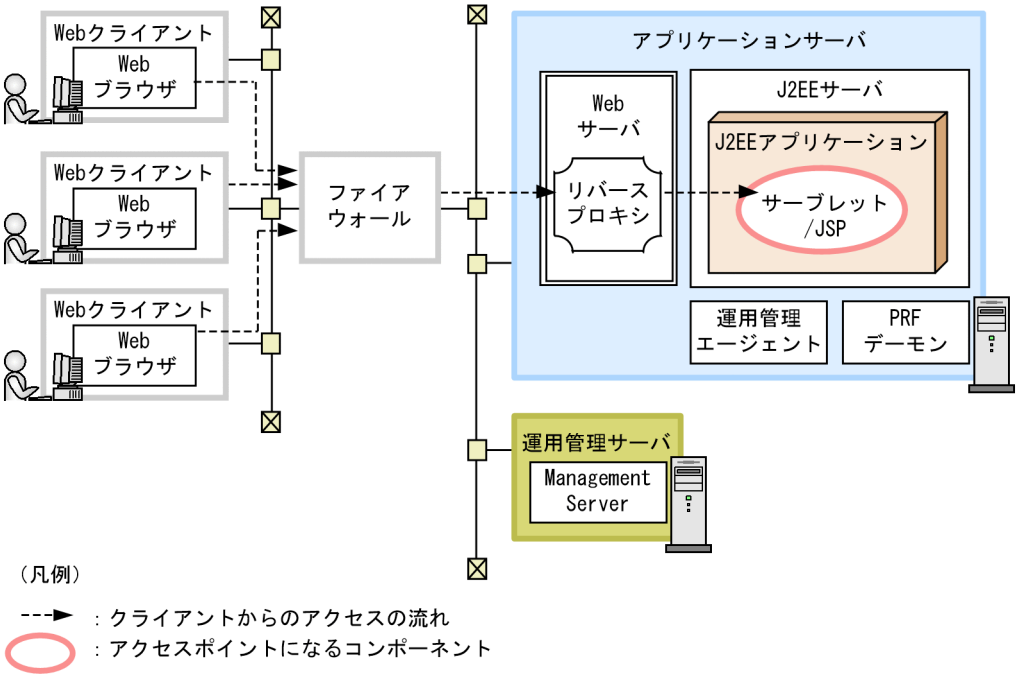
サーブレットと JSP に対するアクセスをファイアウォール経由にする構成について説明します。

(1)　システム構成の特徴

Web クライアント側から見たサーブレットと JSP の手前に，ファイアウォールを配置します。

サーブレットと JSP に対するアクセスをファイアウォール経由にする構成の例を次の図に示します。なお，この図は，Web サーバ連携の場合の例です。

図 3-1　サーブレットと JSP に対するアクセスをファイアウォール経由にする構成



これ以外の例については，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.2　システム構成の説明について」を参照してください。

特徴

サーブレットと JSP に対するアクセスをファイアウォール経由にすることで，システムへの第三者の不正侵入，アプリケーションで扱うデータの漏洩，および第三者による不正な運用を防ぎます。

クライアントからのアクセスの流れ

サーブレットと JSP に対するクライアントからのアクセスは，すべてファイアウォール経由で実行されます。

(2)　それぞれのマシンに必要なソフトウェアと起動するプロセス

ファイアウォールを使用する場合にアプリケーションサーバマシンおよびクライアントマシンに必要なソフトウェアと起動するプロセスは，サーブレットと JSP をアクセスポイントにする構成と同じです。

サーブレットと JSP をアクセスポイントにする構成については，次の個所を参照してください。

- マニュアル「アプリケーションサーバ システム設計ガイド」の「3.4.1　サーブレットと JSP をアクセスポイントに使用する構成（Web サーバ連携の場合)」
- マニュアル「アプリケーションサーバ システム設計ガイド」の「3.4.2　サーブレットと JSP をアクセスポイントに使用する構成（NIO HTTP サーバに直接アクセスする場合)」

3.2.2　Session Bean と Entity Bean に対するファイアウォールの配置

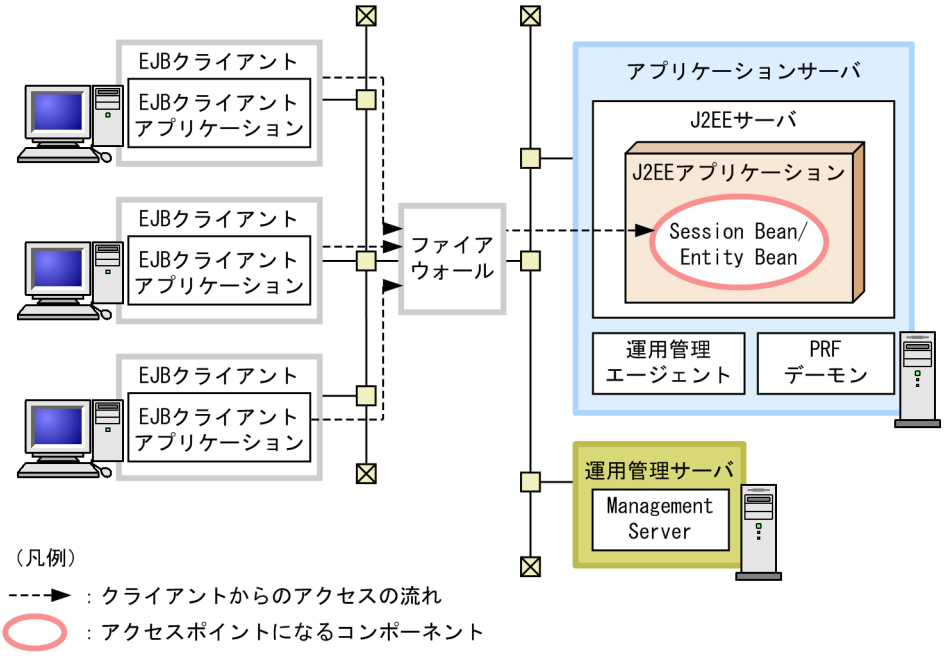
Session Bean と Entity Bean に対するアクセスをファイアウォール経由にする構成です。

(1)　システム構成の特徴

EJB クライアント側から見た Session Bean と Entity Bean の手前に，ファイアウォールを配置します。

Session Bean と Entity Bean に対するアクセスをファイアウォール経由にする構成の例を次の図に示します。

図 3-2　Session Bean と Entity Bean に対するアクセスをファイアウォール経由にする構成



これ以外の凡例については，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.2　システム構成の説明について」を参照してください。

特徴

Session Bean と Entity Bean に対するアクセスをファイアウォール経由にすることで，システムへの第三者の不正侵入，アプリケーションで扱うデータの漏洩，および第三者による不正な運用を防ぎます。

クライアントからのアクセスの流れ

Session Bean と Entity Bean に対する EJB クライアントからのアクセスは，すべてファイアウォール経由で実行されます。

(2)　それぞれのマシンに必要なソフトウェアと起動するプロセス

ファイアウォールを使用する場合にアプリケーションサーバマシンおよびクライアントマシンに必要なソフトウェアと起動するプロセスは，Session Bean と Entity Bean をアクセスポイントにする構成と同じです。マニュアル「アプリケーションサーバ システム設計ガイド」の「3.4.3　Session Bean と Entity Bean をアクセスポイントに使用する構成」を参照してください。

3.2.3　リソースマネジャに対するファイアウォールの配置

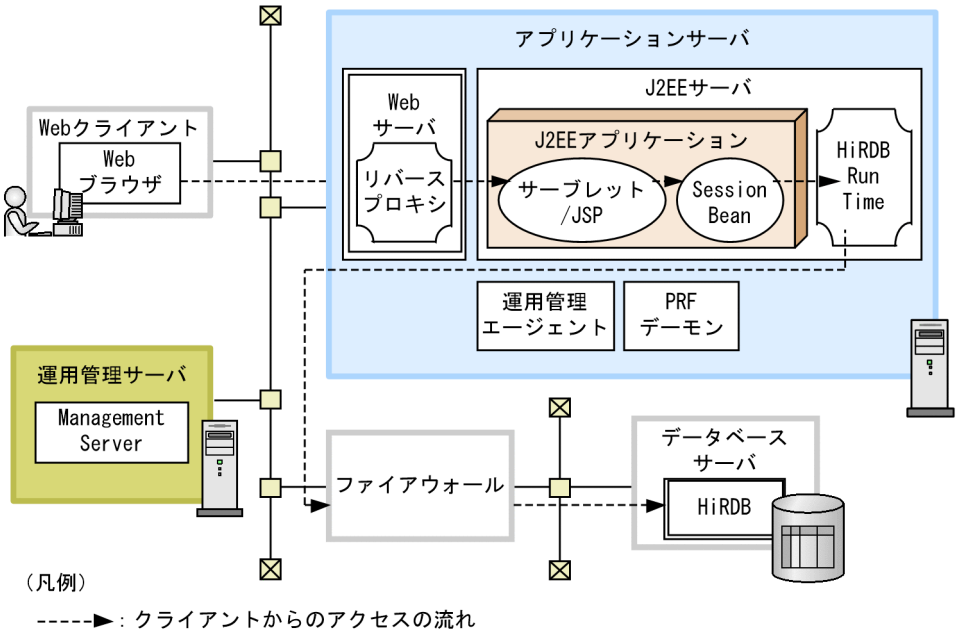
リソースマネジャに対するアクセスをファイアウォール経由にする構成です。

(1)　システム構成の特徴

アプリケーション側から見たリソースマネジャの手前に，ファイアウォールを配置します。

リソースマネジャに対するアクセスをファイアウォール経由にする構成の例を次の図に示します。

図 3-3　リソースマネジャに対するアクセスをファイアウォール経由にする構成



これ以外の凡例については，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.2　システム構成の説明について」を参照してください。

特徴

リソースマネジャに対するアクセスをファイアウォール経由にすることで，システムへの第三者の不正侵入，リソースマネジャで管理するデータの漏洩，および第三者による不正な運用を防ぎます。

クライアントからのアクセスの流れ

クライアントマシン上の Web ブラウザからのリクエストは，Web サーバ経由でサーブレットと JSP に送信されます。サーブレットと JSP は，ローカルで Session Bean を呼び出します。Session Bean からデータベースへのアクセスが，ファイアウォール経由になります。

(2)　それぞれのマシンに必要なソフトウェアと起動するプロセス

トランザクションの使用方法に応じたソフトウェアとプロセスを起動してください。詳細は，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.6　トランザクションの種類を検討する」を参照してください。

3.3　DMZ へのリバースプロキシの配置を検討する

この節では，DMZ にリバースプロキシを配置してセキュリティを確保するための構成について説明します。

インターネットに接続するシステムの場合，ここで説明する構成を基に，リバースプロキシを配置する構成を検討してください。

なお，ここでは，使用する Web サーバの種類に応じた，リバースプロキシの配置について説明します。このほかのセキュリティの考え方については，「4.　セキュアなシステムの検討」を参照してください。

3.3.1　リバースプロキシの配置

NIO HTTP サーバを使用する場合にリバースプロキシを配置する構成について説明します。

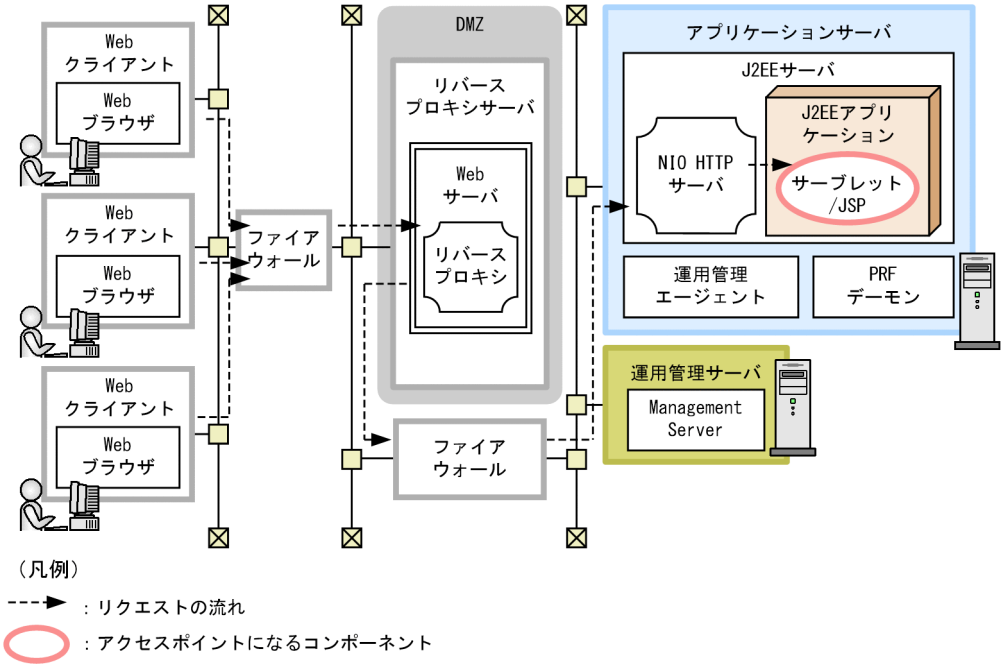
インターネットに接続するシステムで NIO HTTP サーバを使用する場合は，ここで示す構成を基に，必ず DMZ を確保して，リバースプロキシを配置する構成にしてください。

(1)　システム構成の特徴

Web ブラウザとアプリケーションサーバの間に DMZ を確保して，リバースプロキシサーバを配置する構成です。

NIO HTTP サーバを使用する場合に DMZ にリバースプロキシを配置する構成の例を次の図に示します。

図 3-4　NIO HTTP サーバを使用する場合に DMZ にリバースプロキシを配置する構成の例



これ以外の凡例については，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.2　システム構成の説明について」を参照してください。

特徴

- アプリケーションサーバへのアクセスはリバースプロキシサーバからだけになり，Web ブラウザからアプリケーションサーバに直接アクセスされることを抑止できます。
- 通常，リバースプロキシ上には HTML などの静的コンテンツは配置しません。

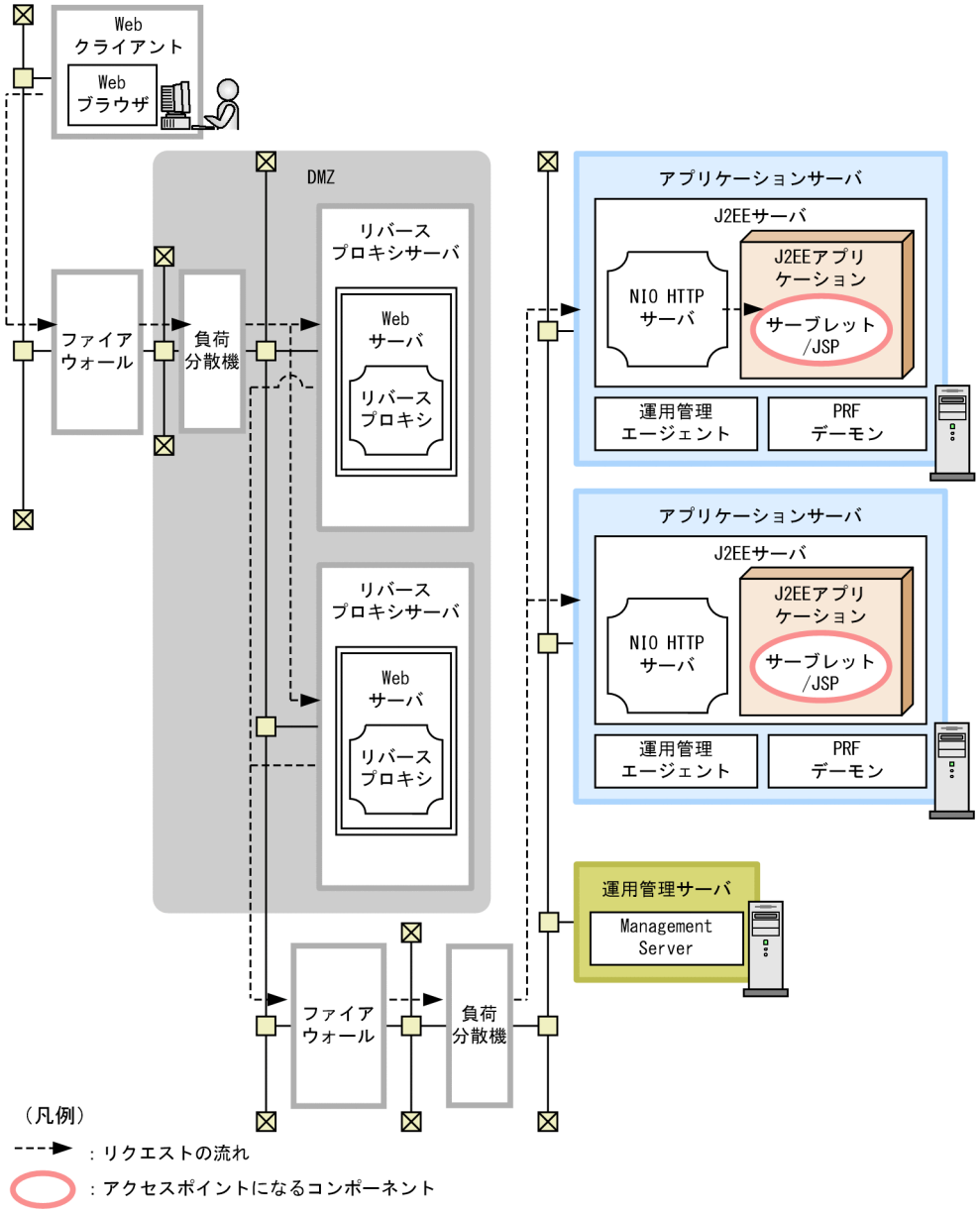
リクエストの流れ

サーブレットと JSP に対するクライアントからのアクセスは，リバースプロキシモジュールを組み込んだ Web サーバ経由で実行されます。

また，ロードバランスクラスタを使用して負荷分散をする場合は，リバースプロキシサーバおよびアプリケーションサーバそれぞれに対して負荷分散機（レイヤ 5 スイッチ）を使用して負荷分散を実行できます。

ロードバランスクラスタ構成の場合に，DMZ にリバースプロキシを配置する構成の例を次の図に示します。

図 3-5　NIO HTTP サーバを使用する場合に DMZ にリバースプロキシを配置する構成の例（ロードバランスクラスタ構成の場合）



これ以外の凡例については，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.2　システム構成の説明について」を参照してください。

特徴

- アプリケーションサーバへのアクセスはリバースプロキシサーバだけからになり，Web ブラウザからアプリケーションサーバに直接アクセスされることを抑止できます。
- 通常，リバースプロキシ上には HTML などの静的コンテンツは配置しません。
- リバースプロキシサーバおよびアプリケーションサーバへの負荷を分散してスケーラビリティと可用性を確保できます。

リクエストの流れ

サーブレットと JSP に対するクライアントからのアクセスは，一つ目の負荷分散機，リバースプロキシモジュールを組み込んだ Web サーバ，および二つ目の負荷分散機経由で実行されます。

このとき，一つ目の負荷分散機は，Web ブラウザからのアクセスを，複数のリバースプロキシサーバに対して振り分けて負荷を分散します。二つ目の負荷分散機は，リバースプロキシサーバからのアクセスを，複数のアプリケーションサーバに対して振り分けて負荷を分散します。二つ目の負荷分散機では，HTTP セッションのスティッキー（Sticky）やアフィニティ（Affinity）の関連づけも制御します。

なお，HTTPS を使用する場合は，一つ目の負荷分散機のフロントに SSL アクセラレータを配置する必要があります。

(2)　それぞれのマシンに必要なソフトウェアと起動するプロセス

それぞれのマシンに必要なソフトウェアとプロセスについて説明します。

(a)　リバースプロキシサーバマシン

リバースプロキシサーバマシンには，HTTP Server をインストールします。

必ず起動するプロセスは次のとおりです。

- Web サーバ

この Web サーバには，リバースプロキシモジュールが組み込まれている必要があります。

(b)　アプリケーションサーバマシン，運用管理サーバマシンおよびクライアントマシン

アプリケーションサーバマシン，運用管理サーバマシンおよびクライアントマシンに必要なソフトウェアと起動するプロセスは，サーブレットと JSP をアクセスポイントにする構成と同じです。マニュアル「アプリケーションサーバ システム設計ガイド」の「3.4.2　サーブレットと JSP をアクセスポイントに使用する構成（NIO HTTP サーバに直接アクセスする場合）」を参照してください。

4 セキュアなシステムの検討

業務システムを安全に稼働させ，扱うデータを確実に守るためには，システム設計の段階でセキュリティについて十分に検討しておく必要があります。この章では，セキュアなシステムを構築・運用するための考え方，正しい構築・運用手順，および監査の方法について説明します。

また，外部ネットワークを使用するシステムの場合に想定されるセキュリティ上の脅威を明確にし，それぞれに対して，ハードウェアおよびソフトウェアを使用して対策する方法についても説明します。

なお，この章は，J2EE アプリケーションを実行するシステムの場合に参照してください。バッチアプリケーションを実行するシステムの場合は該当しません。

4.1　この章の構成

この章では，セキュアなシステムを構築・運用するための考え方，正しい構築・運用手順，および監査の方法について説明します。この章の構成を次の表に示します。

表 4–1　この章の構成（セキュアなシステムの検討）

分類	タイトル	参照先
解説	セキュアなシステムの検討の概要	4.2
	セキュアなシステムの構成の検討	4.3
	システムの使用者の検討	4.4
	システムが扱う資産の検討	4.5
	セキュアなシステムの前提条件の確認	4.6
	想定される脅威の分析	4.7
	対策の検討	4.8
	作業手順の検討	4.9
	システムの監査方法の確認	4.10
	外部ネットワークを使用するシステムでのセキュリティの検討	4.11

注　「実装」,「設定」,「運用」，および「注意事項」について，この章での説明はありません。

4.2　セキュアなシステムの検討の概要

システムを管理または運用するユーザがシステムを構築・運用していく過程や，システムが提供するサービスをエンドユーザが利用していく過程で，システムにはセキュリティ上のさまざまな脅威が想定されます。脅威からシステムを守るには，システムを物理的に安全な構成に設計したり，作業者の運用ルールを定めたりするなど，対策を実施する必要があります。

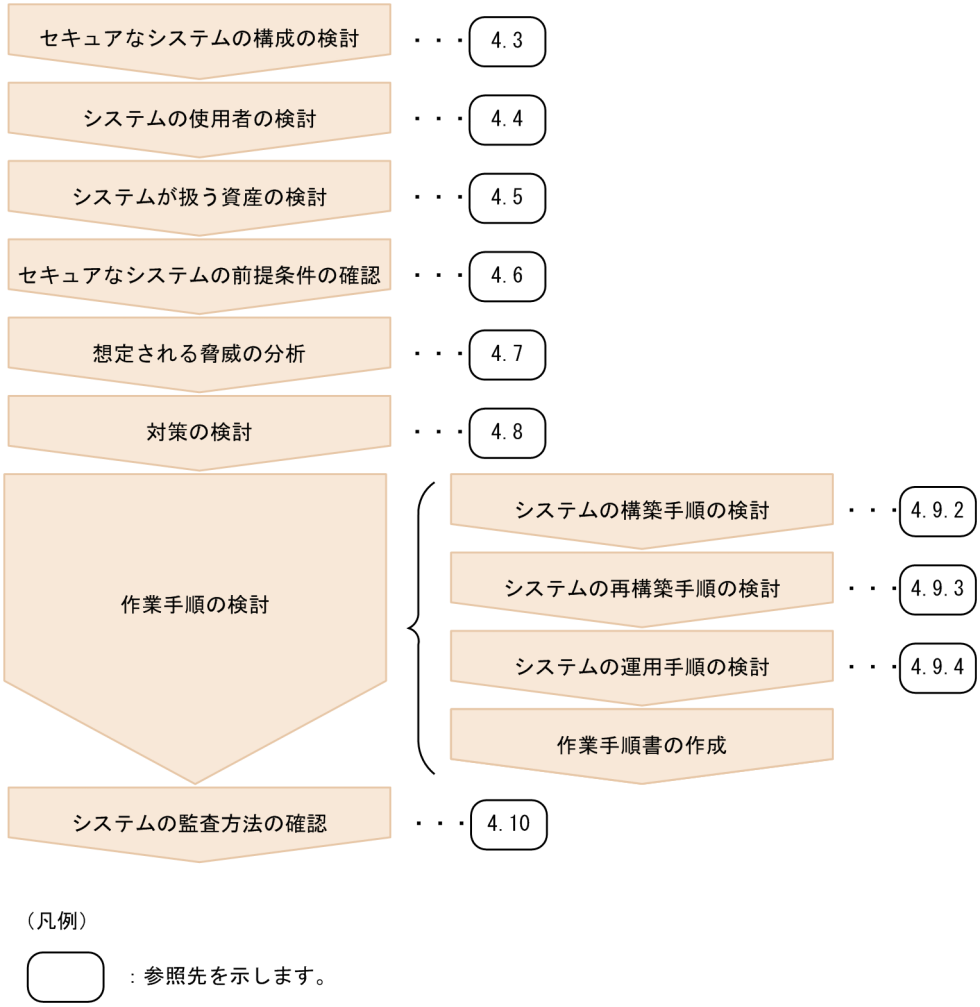
また，近年，組織の健全な運営の保証や，複雑化・多様化する IT システムの安全な構築・運用の観点から，組織の内部統制の重要性が高まっています。その中では，システムがセキュアに保たれていることを監査者に証明できることが求められます。そのためには，システムに対して，いつ，だれが，どんな操作を実施したのかをログに記録して，システムに対して正当な権限を持つ作業者が正当な業務を実施したことを監査できる仕組みが必要です。

このようなセキュアなシステムを実現するためには，想定される脅威をシステムの設計時に明確にして，それに応じた対策を実現できるシステムを検討していく必要があります。

この章では，システムに対して想定される脅威を明確にした上で，セキュアなシステムを構築・運用するための考え方や手順など，システム設計時に検討が必要なことについて説明します。

セキュアなシステムの検討は，次の流れで実施します。

図 4-1　セキュアなシステムの検討の流れ



なお，この図では，企業内部で使用されるシステムのセキュリティを確保するための作業の流れを示しています。外部からの脅威に対する対策については，「4.11　外部ネットワークを使用するシステムでのセキュリティの検討」を参照してください。

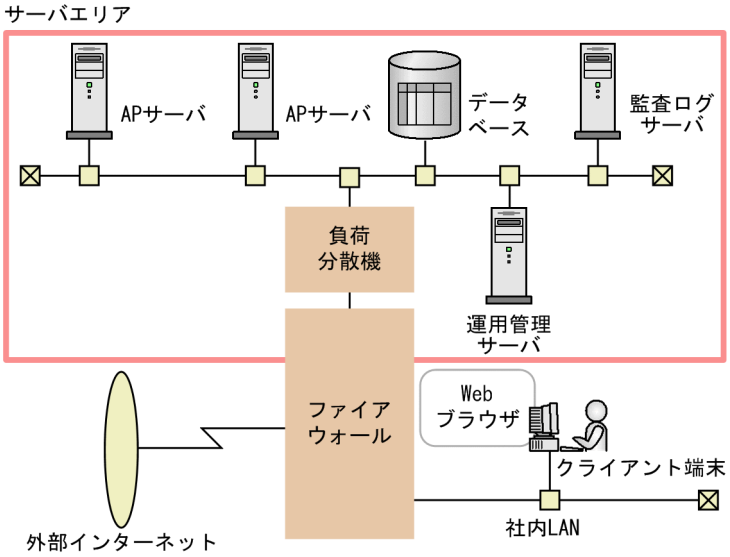
4.3　セキュアなシステムの構成の検討

ここでは，セキュアなシステムの構成について説明します。このマニュアルではセキュアなシステムとして，次のようなシステムを想定しています。

- 大企業で常時稼働して，社内で利用されるシステムです。
- システムを構成する要素はすべて社内 LAN 上に配置されます。
- システムが提供するサービスは，社内のクライアント端末から Web ブラウザを使用して利用されます。
- クライアント端末からサービスを利用するには，ログイン処理が必要です。システムに登録されていないユーザはサービスを利用できません。

このシステムの構成を次の図に示します。

図 4-2　セキュアなシステムの構成



注　APサーバ：アプリケーションサーバ

このシステムの構成要素を次に示します。なお，説明中にある「システム管理者」，「システム運用者」，「監査者」，および「エンドユーザ」の定義については，「4.4　システムの使用者の検討」を参照してください。

サーバエリア

ハードウェアを管理するための，物理的に隔離されたスペースです。サーバエリア内のハードウェアは，システム管理者が管理します。サーバエリアに入室できるのは，システム管理者，システム運用者，および監査者だけです。

アプリケーションサーバ

Web サーバ，サービスを提供する J2EE アプリケーション，および J2EE アプリケーションが稼働するために必要なサーバプログラムが稼働しているマシンです。サーバエリアに複数台設置されて，負荷分散機によって負荷分散されます。

データベース

ユーザの情報やサービスが処理した情報が格納されているマシンです。サーバエリアに設置します。

監査ログサーバ

監査ログを収集して，監査を実施するためのサーバです。監査者だけが使用します。サーバエリアに設置します。

運用管理サーバ

アプリケーションサーバを管理する，運用管理プログラムが稼働するマシンです。システム構築時にシステム管理者が，システム運用時にシステム運用者が使用します。サーバエリアに設置します。

負荷分散機

アプリケーションサーバを複数台設置する場合に，負荷分散をする装置です。サーバエリアに設置します。

ファイアウォール

サーバエリア，社内 LAN，および外部インターネットの間に配置します。

クライアント端末

システムが提供するサービスを利用するための端末です。エンドユーザは，クライアント端末上の Web ブラウザを使用して，社内 LAN 経由でアプリケーションサーバにアクセスします。

4.4　システムの使用者の検討

セキュアなシステムを検討するには，まず，システムの使用者を定義します。システムの使用者にはどのようなユーザがいるかを洗い出して，それぞれの使用者の目的や作業範囲を明確に定義します。これが，システムの監査を実施する観点の一つである，正しいユーザがそれぞれの操作を実行したかどうかを検証する根拠になります。

なお，それぞれの使用者の作業手順は，作業手順書を作成して定義する必要があります。作業手順書には，「システム構築手順書」，「システム運用手順書」，「エンドユーザ操作手順書」，「入退室手順書」などが考えられます。作業手順書の検討については，「4.9　作業手順の検討」を参照してください。

このシステムでは，次に示す使用者を定義します。

システム管理者

システム管理者は，「システム構築手順書」に従って，システムの構築，および管理を実施します。具体的には，主に次の作業を実施します。

- サーバエリア内のハードウェア，ソフトウェア，ネットワークの設置・設定
- ソフトウェアのアップデート
- システムの起動と停止

システム管理者は，組織内の情報システム部門から選ばれたユーザが担当します。

システム運用者

システム運用者は，「システム運用手順書」に従って，サーバエリア内でエンドユーザの登録・削除などのシステムの運用作業を実施します。システム運用者は，社内の情報システム部門から委託されたユーザが担当します。

エンドユーザ

エンドユーザは，「エンドユーザ操作手順書」に従って，システムが提供するサービスを利用します。サービスの利用には，社内 LAN に接続されたクライアント端末上の Web ブラウザを使用します。

監査者

監査者は「入退室手順書」に従ってサーバエリアに入室して，監査ログを取得します。取得した監査ログを調査して，信頼できるシステム管理者が「システム構築手順書」に従った正しい方法でシステムを構築しているかどうかを確認します。また，正しい方法で構築されたシステムが，「システム運用手順書」および「エンドユーザ操作手順書」に従った正しい方法で運用，および利用されているかどうかを監査します。監査者は，社内の内部監査を実施するコンプライアンス部門から選ばれたユーザが担当します。

4.5　システムが扱う資産の検討

セキュアなシステムを検討するには，システムが扱う資産（データ）のうち，保護する必要がある資産にはどのようなものがあるのかを明確に判断する必要があります。

このシステムでは，システムが扱う資産（データ）のうち，次の資産を保護する必要があると判断しています。

- システム管理者のユーザ情報
- エンドユーザのユーザ情報
- システム構築時の設定ファイル
- J2EE アプリケーション
- サービスを利用する中でエンドユーザが送信して，J2EE アプリケーションが処理した情報
- 監査ログ

保護する必要があると判断した資産については，使用できる権限を制御するなどの対策が必要です。対策の詳細については，「4.8　対策の検討」を参照してください。

4.6　セキュアなシステムの前提条件の確認

ここでは，セキュアなシステムの前提条件について説明します。

セキュアなシステムでは，ハードウェアの設置方法や作業者に関する前提条件の確認が必要になります。前提条件を把握した上で，アプリケーションサーバが提供する機能や OS の機能などを使用して，想定される脅威に対する対策を実施します。

ここでは，次の二つの前提条件を想定します。

- 物理的な前提条件
- 運用上の前提条件

4.6.1　物理的な前提条件

セキュアなシステムを構築するための，物理的な前提条件を次に示します。

- システムが稼働するハードウェア，ファイアウォール，それぞれのサーバ，および内部ネットワークが，外部から物理的に隔離されたサーバエリアに設置されていること。
- 施錠管理などによって，認証されたユーザ以外がサーバエリアに入室できないこと。
- システムが稼働するために必要のないハードウェア，およびソフトウェアは，サーバエリア内には持ち込まれないこと。

4.6.2　運用上の前提条件

セキュアなシステムを構築するための運用上の前提条件を次に示します。

運用上の前提条件には，作業手順書，システム管理，システム運用，およびシステム監査に関する前提条件があります。それぞれの前提条件を次に示します。

作業手順書に関する前提条件

システムの構築手順は「システム構築手順書」に，システムの運用手順は「システム運用手順書」に，システムの操作手順は「エンドユーザ操作手順書」に記載されていること。

システム管理に関する前提条件

システムが稼働するために必要なサーバエリア内のハードウェア，ソフトウェア，および J2EE アプリケーションは，「システム構築手順書」に従ってシステム管理者が構築・設定すること。また，システム管理者は，信頼できる者が担当すること。

システム運用に関する前提条件

システムが稼働するために必要な，サーバエリア内のハードウェア，ソフトウェア，および J2EE アプリケーションは，システム運用者が運用すること。

システム監査に関する前提条件

システム監査を実施する監査者は，信頼できる者が担当すること。

4.7　想定される脅威の分析

この節では「4.4　システムの使用者の検討」と「4.5　システムが扱う資産の検討」で検討した内容，および「4.6　セキュアなシステムの前提条件の確認」で確認した内容から，システムにはどんな脅威が想定されるかを分析します。

このシステムでは次の脅威が想定されます。

- **不正なユーザによるサービスの利用**
  システムに登録されていないエンドユーザがサービスを利用するおそれがあります。
- **手順書に従わないユーザによるサービスの利用**
  システムに登録されているユーザ ID およびパスワードを入手しているエンドユーザが，「エンドユーザ操作手順書」に従わないで，システムの脆弱性を利用してサービスを利用するおそれがあります。
  また，システムに登録しているユーザが，権限のないサービスを利用するおそれがあります。
- **不正なシステム管理者によるシステム構築**
  システム管理者ではないユーザが，「入退室手順書」に従わないで不正にサーバエリアに入室してシステムを構築するおそれがあります。
- **不正なシステム運用者によるシステム運用**
  システム運用者ではないユーザが，「入退室手順書」に従わないで不正にサーバエリアに入室してシステムを運用するおそれがあります。
- **手順書に従わないシステム運用者によるシステム運用**
  システム運用者の Management Server の管理ユーザアカウントを使用して，「システム運用手順書」に従わない方法でシステムを運用するおそれがあります。

「4.8　対策の検討」で説明する対策を実施すると，これらの脅威からシステムを守れます。

「入退室手順書」，「システム構築手順書」，「システム運用手順書」，および「エンドユーザ操作手順書」の詳細については，「4.9　作業手順の検討」を参照してください。

4.8　対策の検討

この節では，実施する対策の内容と，対策を実施したシステムの動作について説明します。

実施する対策には，次の二つがあります。

- 前提条件に対して実施する対策
  「4.6　セキュアなシステムの前提条件の確認」で確認した前提条件に対する対策です。
- 想定した脅威に対して実施する対策
  「4.7　想定される脅威の分析」で想定した脅威に対する対策です。

それぞれについて次に示します。

4.8.1　前提条件に対して実施する対策

ここでは，「4.6　セキュアなシステムの前提条件の確認」で確認した前提条件に対して実施する，対策の内容について説明します。

「4.6　セキュアなシステムの前提条件の確認」で確認した前提条件と，実施する対策を次の表に示します。

表 4-2　前提条件と実施する対策

前提条件	対策
物理的な前提条件	• 物理的な対策
運用上の前提条件	• システム管理者に対する対策 • システム運用者に対する対策 • システム監査者に対する対策

それぞれの対策の概要を次に示します。

(1)　物理的な前提条件に対する対策

物理的な前提条件に対する対策を次に示します。

- **物理的な対策**
  - システム管理者は，システムが稼働するハードウェア，ファイアウォール，それぞれのサーバ，および内部ネットワークを，外部から物理的に隔離されたサーバエリアに設置します。
  - システム管理者は，システムが稼働するために必要のないハードウェア，およびソフトウェアを，サーバエリア内に持ち込まないようにします。
  - システム管理者，システム運用者，および監査者は，「入退室手順書」に従ってサーバエリアに入退室するようにします。

「入退室手順書」については，「4.9　作業手順の検討」を参照してください。

(2)　運用上の前提条件に対する対策

運用上の前提条件に対する対策を次に示します。

- システム管理者に対する対策
  - システム管理者には，システム全体に責任を持っていて，悪意のある行為はしない，信頼できるユーザを選定すること。
  - システム管理者は，システムの構築および管理に関するトレーニングなどを実施して，システムの構築，および管理方法を熟知していること。また，システムで利用するそれぞれの機器の構築，および管理方法について熟知していること。
  - システム管理者は，セキュリティ面での注意点を考慮しながらシステムを構築および管理すること。
  - システム管理者は，自分の OS のパスワード，システム管理者の Management Server の管理パスワード，システム運用者の OS のパスワード，およびシステム運用者の Management Server の管理パスワードとして，推測されにくい十分に強度のあるパスワードを設定すること。

  「システム構築手順書」については，「4.9　作業手順の検討」を参照してください。
- システム運用者に対する対策
  - システム運用者は，システムの運用に関するトレーニングなどを実施して，システムの運用方法を熟知していること。
  - システム運用者は，セキュリティ面での注意点を考慮しながらシステムを運用すること。
  - システム運用者は，エンドユーザのパスワードとして，推測されにくい十分に強度のあるパスワードを設定すること。

  「システム運用手順書」については，「4.9　作業手順の検討」を参照してください。
- システム監査者に対する対策
  - 監査者には，システム全体に責任を持っていて，悪意のある行為はしない，信頼できるユーザを選定すること。
  - 監査者には，システム管理者以外のユーザを選定すること。
  - 監査者は，システムの構築手順が正当であるかどうかを確認すること。また，運用手順が正当であるかどうかを監査すること。

4.8.2　想定した脅威に対して実施する対策

ここでは，「4.7　想定される脅威の分析」で想定した脅威に対して実施する，対策の内容について説明します。

このシステムで想定される脅威と，それぞれの脅威に対する対策を，対策を実施する対象者ごとに次の表に示します。それぞれの脅威の詳細については，「4.7　想定される脅威の分析」を参照してください。

表 4-3　想定される脅威と実施する対策

対策の対象者	脅威	対策
システム管理者	不正なシステム管理者によるシステム構築	• OS のユーザ識別・認証
システム運用者	不正なシステム運用者によるシステム運用	• OS のユーザ識別・認証 • システム運用者のユーザ識別・認証
	手順書に従わないシステム運用者によるシステム運用	• システムの監査ログ出力 • J2EE アプリケーションの監査ログ出力
エンドユーザ	不正なユーザによるサービスの利用	• J2EE アプリケーションの監査ログ出力 • J2EE アプリケーションのユーザ識別・認証
	手順書に従わないユーザによるサービスの利用	• J2EE アプリケーションの監査ログ出力 • J2EE アプリケーションのアクセス制御

それぞれの対策の概要を次に示します。

システム管理者を対象にした対策

- **OS のユーザ識別・認証**

  システム管理者だけがシステムを管理するように，システムが動作する OS のユーザ識別・認証を設定して，コマンドの実行権限を制御します。

システム運用者を対象にした対策

- **OS のユーザ識別・認証**

  システム運用者がシステムを運用するように，システムが動作する OS のユーザ識別・認証を設定して，コマンドの実行権限を制御します。

- **システム運用者のユーザ識別・認証**

  システム運用者がシステムを運用するように，システムでユーザ識別・認証をします。

- **システムの監査ログ出力**

  手順書に従った方法でシステムを運用したかどうかを監査するために，システムで監査ログを出力するようにします。

- **J2EE アプリケーションの監査ログ出力**

  手順書に従った方法でエンドユーザを管理したかどうかを監査するために，アプリケーションサーバが提供する監査ログ出力用の API を使用して J2EE アプリケーションを実装して，J2EE アプリケーションの監査ログを出力するようにします。監査ログ出力用の API を使用した J2EE アプリケーションの実装方法については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「6.8　アプリケーションの監査ログを出力するための実装」を参照してください。

エンドユーザを対象にした対策

- **J2EE アプリケーションの監査ログ出力**

  正当なエンドユーザが手順書に従った方法でサービスを利用したかどうかを監査するために，アプリケーションサーバが提供する監査ログ出力用の API を使用して J2EE アプリケーションを実装し

て，J2EE アプリケーションの監査ログを出力するようにします。監査ログ出力用の API を使用した J2EE アプリケーションの実装方法については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「6.8　アプリケーションの監査ログを出力するための実装」を参照してください。

- **J2EE アプリケーションのユーザ識別・認証**

  正当なエンドユーザだけがサービスを利用するように，J2EE アプリケーションにユーザ識別・認証機能を実装します。

- **J2EE アプリケーションのアクセス制御**

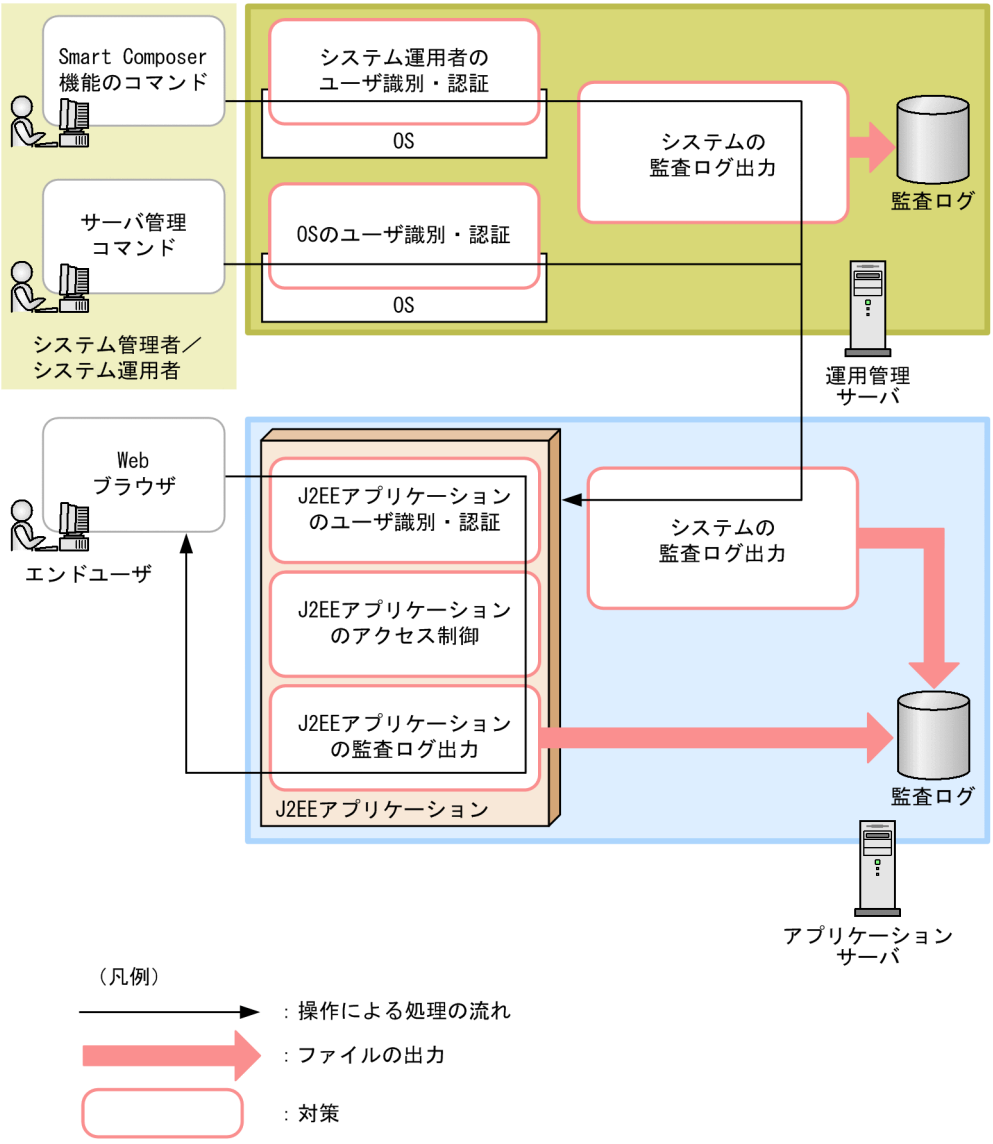
  アクセス権限を持っているエンドユーザだけが保護するべきデータにアクセスできるように，J2EE アプリケーションにアクセス制御機能を実装します。

4.8.3　対策を実施したセキュアなシステムの動作

ここでは，対策を実施したセキュアなシステムの動作について説明します。

対策を実施したシステムの動作の概要を次の図に示します。なお，図中に示した対策は，「4.8.2　想定した脅威に対して実施する対策」で説明した対策と対応しています。

図 4-3　システム管理者の操作とシステムの動作



この図で示した，対策を実施したシステムの動作の概要を，システムの使用者ごとに説明します。

(1)　システム管理者・システム運用者の操作とシステムの動作

システム管理者・システム運用者の操作とシステムの動作の概要について説明します。

システム管理者の操作

- Smart Composer 機能のコマンドを使用してアプリケーションサーバを構築します。ただし，J2EE アプリケーションの設定，リソースの設定などの作業は，サーバ管理コマンドを使用して実施します。
- ユーザ識別・認証機能とアクセス制御機能，および監査ログを出力する機能を実装したアプリケーションを，デプロイ・開始します。

システム運用者の操作

Smart Composer 機能のコマンドを使用してシステムを運用します。ただし，障害発生時のログの収集は，snapshotlog コマンドを使用して実施します。

システムの動作

コマンドの実行によるそれぞれの操作に対して監査ログを出力します。

ポイント

一部のコマンドは監査ログを出力しないため，使用するコマンドが監査ログを出力するコマンドかどうかを確認して作業を実施してください。監査ログを出力するコマンドについては，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「6.6　監査ログを出力するコマンド・操作一覧」を参照してください。

(2)　エンドユーザの操作とシステムの動作

システムが提供するサービス利用時のエンドユーザの操作とシステム（J2EE アプリケーション）の動作の概要を次に示します。

エンドユーザの操作

クライアント端末の Web ブラウザから HTTP リクエストをアプリケーションサーバへ送信します。

システム（J2EE アプリケーション）の動作

- HTTP リクエストに含まれているユーザ情報を識別して，ユーザ認証をします。
- アクセス制御機能を使用して，認証されたユーザにアクセス権があるかどうかをチェックします。
- アクセス制御機能で許可されたリクエストが J2EE アプリケーションのサービスを実行します。
- 処理の実行時に，監査ログを出力します。

4.9　作業手順の検討

この節では，セキュアなシステムを構築・運用するために，システムごとに検討する必要がある作業手順について説明します。

セキュアなシステムを構築・運用するには，それぞれの作業者が実施する作業の手順を明確にする必要があります。システムの監査では，監査ログと作業手順書を照合して，それぞれに矛盾がないかを調査します。**作業手順書**とは，システム構築，システム運用，エンドユーザによるサービスの利用などの作業について，正しい手順，方法を明文化した文書です。作業手順書は，それぞれのシステムに応じて作成する必要があります。作業者や操作内容の記録である監査ログと作業手順書を照合することで，正しい作業者が正しい手段・手順で作業を実施したかどうかが明確になります。これによって，システムの安全性を保てます。

作業手順書の作成では，それぞれの作業手順書に記載する作業とその手順・方法を検討する必要があります。監査の観点で，正しいユーザが正しい手順・方法で実施するように規定する必要がある作業を洗い出してください。また，それぞれの作業の手順では，必ず監査ログを出力するコマンドを使用して作業を実行するように規定してください。

4.9.1　作成する作業手順書の概要

ここでは，作成する必要がある作業手順書の概要について説明します。

作成する作業手順書は，システムによって異なります。このシステムでは，次の作業手順書を作成します。

- **入退室手順書**

  施錠管理がされているサーバエリアに入退室する手順が規定された文書です。

- **システム構築手順書**

  システムを構築する手順が規定された文書です。記載する手順は，「4.9.2　システムの構築手順の検討」および「4.9.3　システムの再構築手順の検討」で説明している手順を基に作成します。

  なお，「4.9.2　システムの構築手順の検討」，および「4.9.3　システムの再構築手順の検討」で説明している手順では，システム管理者は Smart Composer 機能のコマンドおよびサーバ管理コマンドを使用して操作を実行しています。また，すべての操作に，監査ログが出力されるコマンドを使用しています。

- **システム運用手順書**

  システムを運用する手順が規定された文書です。記載する手順は，「4.9.4　システムの運用手順の検討」で説明している手順を基に作成します。

  なお，「4.9.4　システムの運用手順の検討」で説明している手順では，システム運用者は Smart Composer 機能のコマンド，および snapshotlog コマンドを使用して操作を実行しています。

- **エンドユーザ操作手順書**

  システムが提供するサービスを利用する手順が規定された文書です。

4.9.2　システムの構築手順の検討

ここでは，「システム構築手順書」に記載するシステムの構築手順の例について説明します。「システム構築手順書」を作成するときは，ここで説明している手順を参考にしてください。

セキュアなシステムの構築には，Smart Composer 機能のコマンド，およびサーバ管理コマンドを使用します。また，すべての操作に，実行時に監査ログが出力されるコマンドを使用します。ここで説明している以外の操作を作業手順書に記載する場合も，監査ログが出力されるコマンドを使用する方法を記載してください。監査ログが出力されるコマンドについては，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「6.6　監査ログを出力するコマンド・操作一覧」を参照してください。

なお，この項で説明する手順は，すべてシステム管理者が実行します。

(1)　ハードウェアの設置

システム管理者が，ハードウェアを設置します。ハードウェアを設置する手順を次に示します。

1. 「入退室手順書」に従って，外部から物理的に隔離されたサーバエリアに入室します。
2. システムが稼働するハードウェア，およびファイアウォールをサーバエリア内に設置します。

なお，「システム構築手順書」では，ハードウェア，およびファイアウォールを設置する手順の詳細を規定してください。

(2)　OS のインストール

システム管理者が，システムで使用する OS のインストールを実施します。OS をインストールする手順を次に示します。

1. OS のインストール，および IP アドレス・ホスト名の設定などの，ネットワークの接続に必要な設定をします。
2. システムに必要なセキュリティパッチを適用します。
3. システムに必要なソフトウェアをインストールして，環境変数を設定します。
4. システム管理者用の OS のアカウントを作成して，管理者権限を付与します。
5. 監査者用の OS のアカウントを作成して，安全な手段で監査者へ通知します。

(3)　システムの管理の開始

システム管理者が，「(2) OS のインストール」で設定したシステム管理者用の OS のアカウントを使用して OS にログインします。

(4) 監査ログを出力するための設定

システム管理者が，運用管理サーバおよびアプリケーションサーバの端末で監査ログを出力するための設定をします。監査ログを出力するための設定をする手順を次に示します。

1. システム構成を基に，ログファイルのサイズを決定します。
2. システム管理者，およびシステム運用者に，監査ログファイルに対する読み取り権限と書き込み権限を設定します。また，監査者に監査ログファイルに対する読み取り権限を設定します。
3. 手順 1.および手順 2.で決定，および設定した内容を監査ログ定義ファイル（auditlog.properties）に反映します。
4. 監査ログ定義ファイルに指定した監査ログ出力ディレクトリを作成します。
5. システム管理者，およびシステム運用者に，手順 4.で作成した監査ログ出力ディレクトリに対する読み取り権限と書き込み権限を設定します。また，監査者に，手順 4.で作成した監査ログ出力ディレクトリに対する読み取り権限を設定します。
6. セットアップコマンド（auditsetup コマンド）を実行します。

(5) 負荷分散機およびデータベースの設定

システム管理者が，負荷分散機およびデータベースをサーバエリア内に設置して，負荷分散機およびデータベースを設定します。

なお，「システム構築手順書」では，負荷分散機およびデータベースの設定手順の詳細を規定してください。

(6) 運用管理サーバの設定

システム管理者が，運用管理サーバの初期設定をします。運用管理サーバを設定する手順を次に示します。

1. mngsvrctl コマンドで，引数「setup」を指定して，Management Server のセットアップ，および Management Server の管理ユーザアカウントの設定をします。
2. mngautorun コマンドで，引数「server」と-sync オプションを指定して，Management Server の自動起動の設定をします。

(7) Web システムの構成定義

システム管理者が，Web システムの構成を定義します。Web システムの構成を定義する手順を次に示します。

1. mngsvrctl コマンドで，引数「start」と-sync オプションを指定して，Management Server を起動します。
2. 簡易構築定義ファイルを編集して保存します。

3. adminagentctl コマンドで引数「start」と-sync オプションを指定して，それぞれのアプリケーションサーバで運用管理エージェントを起動します。

4. 運用管理サーバで cmx_build_system コマンドを使用して，Web システムを構築します。

(8)　Web システムの準備

システム管理者が，運用管理サーバの管理者端末で Smart Composer 機能のコマンドを使用して，Web システムの準備をします。Web システムを準備する手順を次に示します。

1. cmx_start_target コマンドを使用して，Web システムを準備状態にします。

2. cmx_list_status コマンドを使用して，Web システム内のサービスユニットが準備状態であることを確認します。

(9)　リソースアダプタの設定

システム管理者が，運用管理サーバの管理者端末でサーバ管理コマンドを使用して，データベースと連携するアプリケーションに必要なリソースアダプタを設定します。リソースアダプタを設定する手順を次に示します。

1. 次のディレクトリから，使用するリソースアダプタの Connector 属性ファイルのテンプレートをコピーします。

   Windows の場合

   <Application Server のインストールディレクトリ>¥CC¥admin¥templates¥

   UNIX の場合

   /opt/Cosminexus/CC/admin/templates/

2. 手順 1.でコピーしてきた Connector 属性ファイルのテンプレートを編集します。

3. cjimportres コマンドを使用して，リソースアダプタをインポートします。

4. cjsetresprop コマンドを使用して，Connector 属性ファイルの編集内容をリソースアダプタに反映します。

5. cjdeployrar コマンドを使用して，リソースアダプタをデプロイします。

6. cjtestres コマンドを使用して，リソースアダプタの接続テストを実施します。

(10)　J2EE アプリケーションの確認

システム管理者が，「4.8.2　想定した脅威に対して実施する対策」で説明した対策が J2EE アプリケーションに施されているかを確認します。ここでは，次の対策について確認します。

- J2EE アプリケーションの監査ログ出力

- J2EE アプリケーションのユーザ識別・認証
- J2EE アプリケーションのアクセス制御

具体的には，J2EE アプリケーションが次の仕様を満たしているかどうかを確認してください。

- システム運用者によってエンドユーザのユーザ ID・パスワードを登録，および削除するための機能がある。
- ユーザ ID・パスワードの識別・認証機能がある。
- 提供するサービスに対するアクセス制御機能がある。
- ユーザがサービスを利用するときに，監査ログを出力する機能がある。

(11)　J2EE アプリケーションの設定

システム管理者が，運用管理サーバの管理者端末でサーバ管理コマンドを使用して，J2EE アプリケーションを設定します。J2EE アプリケーションを設定する手順を次に示します。

1. cjimportapp コマンドを使用して，J2EE アプリケーションをインポートします。
2. cjgetappprop コマンドを使用して，アプリケーション統合属性ファイルを取得します。
3. 手順 2.で取得したアプリケーション統合属性ファイルを編集します。
4. cjsetappprop コマンドを使用して，アプリケーション統合属性ファイルの編集内容を J2EE アプリケーションに反映します。

注意事項

ここでは，実行時情報を持たない J2EE アプリケーションを設定する手順について説明しています。実行時情報を持つ J2EE アプリケーションを設定する場合は，手順 1.で J2EE アプリケーションをインポートしたあとに，cjstopapp コマンドを使用して J2EE アプリケーションを停止してから，手順 2.に進んでください。

(12)　Web システムの開始

システム管理者が，運用管理サーバの管理者端末で Smart Composer 機能のコマンドとサーバ管理コマンドを使用して，Web システムを開始します。Web システムを開始する手順を次に示します。

1. cjstartrar コマンドを使用して，リソースアダプタを開始します。
2. cjstartapp コマンドを使用して，J2EE アプリケーションを開始します。
3. cmx_start_target コマンドを使用して，Web システム内のサービスユニットを稼働状態にします。

(13) 不要な機能の無効化

不正なユーザによって不要な機能を使用されないように，機能を無効化します。具体的には，システム管理者が，コマンドの実行権限を変更したり，コマンドの実行に必要なファイルを削除したりします。無効化する必要がある機能を，Windows の場合と UNIX の場合に分けて，次の表に示します。

表 4-4 無効化する必要がある機能（Windows の場合）

機能名	対象ディレクトリ	対象ファイル	対処
HTTP Server の GUI サーバ管理機能	<Application Server のインストールディレクトリ>¥httpsd	adm-httpsd.exe	システム管理者以外の実行権限を解除します。
HTTP Server のパスワードファイル編集コマンド	<Application Server のインストールディレクトリ>¥httpsd¥bin	htpasswd.exe	システム管理者以外の実行権限を解除します。
CTM のスケジュールキューの同時実行数の変更	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmchpara.exe	システム管理者以外の実行権限を解除します。
CTM の CTM ドメインの情報表示と削除	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmdminfo.exe	システム管理者以外の実行権限を解除します。
CTM のスケジュールキューの閉塞	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmholdque.exe	システム管理者以外の実行権限を解除します。
CTM の実行形式ファイルおよびライブラリのバージョン情報の出力	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmjver.exe	システム管理者以外の実行権限を解除します。
CTM のメッセージの編集と出力	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmlogcat.exe	システム管理者以外の実行権限を解除します。
CTM のスケジュールキュー情報の出力	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmlsque.exe	システム管理者以外の実行権限を解除します。
CTM のスケジュールキューの閉塞解除	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmrlesque.exe	システム管理者以外の実行権限を解除します。
CTM の稼働統計情報の編集と出力	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmstsed.exe	システム管理者以外の実行権限を解除します。
CTM のバッファ内容の強制ファイル出力	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmstsflush.exe	システム管理者以外の実行権限を解除します。
CTM の実行形式ファイルおよびライブラリのバージョン情報の出力	<Application Server のインストールディレクトリ>¥CTM¥bin	ctmver.exe	システム管理者以外の実行権限を解除します。
PRF の性能解析トレース情報の編集出力	<Application Server のインストールディレクトリ>¥PRF¥bin	cprfed.exe	システム管理者以外の実行権限を解除します。
PRF のバッファ内容の強制ファイル出力	<Application Server のインストールディレクトリ>¥PRF¥bin	cprfflush.exe	システム管理者以外の実行権限を解除します。

機能名	対象ディレクトリ	対象ファイル	対処
PRF トレース取得レベルの表示と変更	<Application Server のインストールディレクトリ>¥PRF¥bin	cprflevel.exe	システム管理者以外の実行権限を解除します。
Management Server で使用するコマンド	<Application Server のインストールディレクトリ>¥manager¥bin	mngsvrutil.exe	システム管理者以外の実行権限を解除します。
	<Application Server のインストールディレクトリ>¥manager¥bin	mstrexport.exe	システム管理者以外の実行権限を解除します。
	<Application Server のインストールディレクトリ>¥manager¥bin	mstrimport.exe	システム管理者以外の実行権限を解除します。
	<Application Server のインストールディレクトリ>¥manager¥bin	ssoexport.exe	システム管理者以外の実行権限を解除します。
	<Application Server のインストールディレクトリ>¥manager¥bin	ssogenkey.exe	システム管理者以外の実行権限を解除します。
	<Application Server のインストールディレクトリ>¥manager¥bin	ssoimport.exe	システム管理者以外の実行権限を解除します。
	<Application Server のインストールディレクトリ>¥manager¥bin	uachpw.exe	システム管理者以外の実行権限を解除します。
	<Application Server のインストールディレクトリ>¥manager¥bin	mngsvr_adapter_setup.exe	このコマンドを実行しないようにします。
	<Application Server のインストールディレクトリ>¥manager¥bin	Adapter_HITACHI_COSMINEXUS_MANAGER.exe	システム管理者以外の実行権限を解除します。
	<Application Server のインストールディレクトリ>¥manager¥externals¥jp1¥mngsvrmonitor	mngsvr_monitor_setup.exe	このコマンドを実行しないようにします。
運用管理ポータル	<Application Server のインストールディレクトリ>¥manager¥containers¥m¥webapps¥mngsvr	index.jsp	ファイルを削除します。
	<Application Server のインストールディレクトリ>¥manager¥containers¥m¥webapps¥mngsvr	login.jsp	ファイルを削除します。

表 4-5　無効化する必要がある機能（UNIX の場合）

機能名	対象ディレクトリ	対象ファイル	対処
HTTP Server の GUI サーバ管理機能	/opt/hitachi/httpsd/sbin	adminctl	システム管理者以外の実行権限を解除します。
	/opt/hitachi/httpsd/sbin	adm-httpsd	システム管理者以外の実行権限を解除します。

機能名	対象ディレクトリ	対象ファイル	対処
HTTP Server のパスワードファイル編集コマンド	/opt/hitachi/httpsd/bin	htpasswd	システム管理者以外の実行権限を解除します。
CTM のスケジュールキューの同時実行数の変更	/opt/Cosminexus/CTM/bin	ctmchpara	システム管理者以外の実行権限を解除します。
CTM の CTM ドメイン情報の表示と削除	/opt/Cosminexus/CTM/bin	ctmdminfo	システム管理者以外の実行権限を解除します。
CTM のスケジュールキューの閉塞	/opt/Cosminexus/CTM/bin	ctmholdque	システム管理者以外の実行権限を解除します。
CTM の実行形式ファイルおよびライブラリのバージョン情報の出力	/opt/Cosminexus/CTM/bin	ctmjver	システム管理者以外の実行権限を解除します。
CTM のメッセージの編集と出力	/opt/Cosminexus/CTM/bin	ctmlogcat	システム管理者以外の実行権限を解除します。
CTM のスケジュールキュー情報の出力	/opt/Cosminexus/CTM/bin	ctmlsque	システム管理者以外の実行権限を解除します。
CTM のスケジュールキューの閉塞解除	/opt/Cosminexus/CTM/bin	ctmrlesque	システム管理者以外の実行権限を解除します。
CTM の稼働統計情報の編集と出力	/opt/Cosminexus/CTM/bin	ctmstsed	システム管理者以外の実行権限を解除します。
CTM のバッファ内容の強制ファイル出力	/opt/Cosminexus/CTM/bin	ctmstsflush	システム管理者以外の実行権限を解除します。
CTM の実行形式ファイルおよびライブラリのバージョン情報の出力	/opt/Cosminexus/CTM/bin	ctmver	システム管理者以外の実行権限を解除します。
PRF の性能解析トレース情報の編集出力	/opt/Cosminexus/PRF/bin	cprfed	システム管理者以外の実行権限を解除します。
PRF のバッファ内容の強制ファイル出力	/opt/Cosminexus/PRF/bin	cprfflush	システム管理者以外の実行権限を解除します。
PRF トレース取得レベルの表示と変更	/opt/Cosminexus/PRF/bin	cprflevel	システム管理者以外の実行権限を解除します。
Management Server で使用するコマンド	/opt/Cosminexus/manager/bin	mngsvrutil	システム管理者以外の実行権限を解除します。
	/opt/Cosminexus/manager/bin	mstrexport	システム管理者以外の実行権限を解除します。
	/opt/Cosminexus/manager/bin	mstrimport	システム管理者以外の実行権限を解除します。

機能名	対象ディレクトリ	対象ファイル	対処
	/opt/Cosminexus/manager/bin	ssoexport	システム管理者以外の実行権限を解除します。
	/opt/Cosminexus/manager/bin	ssogenkey	システム管理者以外の実行権限を解除します。
	/opt/Cosminexus/manager/bin	ssoimport	システム管理者以外の実行権限を解除します。
	/opt/Cosminexus/manager/bin	uachpw	システム管理者以外の実行権限を解除します。
	/opt/Cosminexus/manager/bin	mngsvr_adapter_setup	システム管理者以外の実行権限を解除します。
	/opt/Cosminexus/manager/bin	Adapter_HITACHI_COSMINEXUS_MANAGER	システム管理者以外の実行権限を解除します。
運用管理ポータル	/opt/Cosminexus/manager/containers/m/webapps/mngsvr	index.jsp	ファイルを削除します。
	/opt/Cosminexus/manager/containers/m/webapps/mngsvr	login.jsp	ファイルを削除します。

(14)　システム運用者の登録

システム管理者が，運用管理サーバの管理者端末で OS の機能，および Smart Composer 機能のコマンドを使用して，システム運用者のユーザ ID とパスワードを設定します。また，設定したユーザ ID とパスワードをシステム運用者に通知します。システム運用者を登録する手順を次に示します。

1. OS の機能を使用して，システム運用者の OS のユーザ ID とパスワードを設定します。
2. OS の機能を使用して，システム運用者に管理者権限を与えないよう設定します。
3. cmx_admin_passwd コマンドを使用して，システム管理者の Management Server の管理ユーザ ID と管理パスワードを，システム運用者の Management Server の管理ユーザ ID と管理パスワードに変更します。
4. 手順 1.および手順 3.で設定したユーザ ID とパスワードを安全な手段でシステム運用者へ通知します。

4.9.3　システムの再構築手順の検討

ここでは，「システム構築手順書」に記載するシステムの再構築手順の例について説明します。「システム構築手順書」を作成するときは，ここで説明している手順を参考にしてください。

セキュアなシステムの再構築には，Smart Composer 機能のコマンド，およびサーバ管理コマンドを使用します。また，すべての操作に，実行時に監査ログが出力されるコマンドを使用します。ここで説明して

いる以外の操作を作業手順書に記載する場合も，監査ログが出力されるコマンドを使用する方法を記載してください。監査ログが出力されるコマンドについては，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「6.6　監査ログを出力するコマンド・操作一覧」を参照してください。

なお，この項で説明する手順は，すべてシステム管理者が実行します。

(1)　J2EE アプリケーションの入れ替え

システム管理者が，メンテナンスが必要な場合などに J2EE アプリケーションを入れ替えます。J2EE アプリケーションを入れ替える手順を次に示します。

1. cmx_stop_target コマンドを使用して，Web システム内のサービスユニットを準備状態にします。
2. cjstopapp コマンドを使用して，入れ替え前の J2EE アプリケーションを停止します。
3. cjdeleteapp コマンドを使用して，入れ替え前の J2EE アプリケーションを削除します。
4. cjimportapp コマンドを使用して，入れ替え後の J2EE アプリケーションをインポートします。
5. cjgetappprop コマンドを使用して，入れ替え後の J2EE アプリケーションのアプリケーション統合属性ファイルを取得します。
6. 手順 5.で取得したアプリケーション統合属性ファイルを編集して，J2EE アプリケーションに必要な情報を設定します。また，必要に応じて J2EE アプリケーションをカスタマイズします。
7. cjsetappprop コマンドを使用して，手順 6.で編集したアプリケーション統合属性ファイルを，入れ替え後の J2EE アプリケーションに反映します。
8. cjstartapp コマンドを使用して，入れ替え後の J2EE アプリケーションを開始します。
9. cmx_start_target コマンドを使用して，Web システム内のサービスユニットを稼働状態にします。

なお，J2EE アプリケーションの入れ替えには，この手順のほかに，cjreplaceapp コマンドを使用したリデプロイ機能，または cjreloadapp コマンドを使用したリロード機能を使用することもできます。

(2)　システムのチューニング

システム管理者が，必要に応じてシステムをチューニングします。システムをチューニングする手順を次に示します。

1. 簡易構築定義ファイルを編集します。
2. cmx_stop_target コマンドを使用して，Web システム内のサービスユニットを停止します。
3. cmx_build_system コマンドを使用して，Web システムの設定を変更します。
4. cmx_start_target コマンドを使用して，Web システム内のサービスユニットを起動します。

(3)　システム構成の変更（サービスユニットの追加）

システム管理者が，必要に応じてサービスユニットを追加してシステム構成を変更します。サービスユニットを追加してシステム構成を変更する手順を次に示します。

1. 構成変更定義ファイルを作成・編集します。
2. cmx_change_model コマンドを使用して，Management Server の Web システムの情報モデルを変更します。
3. cmx_build_system コマンドを使用して，変更した Web システムの情報モデルを適用します。
4. cmx_start_target コマンドを使用して，追加する Web システム内のサービスユニットを準備状態にします。
5. cjstartrar コマンドを使用して，リソースアダプタを開始します。
6. cjstartapp コマンドを使用して，J2EE アプリケーションを開始します。
7. cmx_start_target コマンドを使用して，追加する Web システム内のサービスユニットを稼働状態にします。

(4)　システム構成の変更（サービスユニットの削除）

システム管理者が，必要に応じてサービスユニットを削除してシステム構成を変更します。サービスユニットを削除してシステム構成を変更する手順を次に示します。

1. cmx_stop_target コマンドを使用して，削除する Web システム内のサービスユニットを停止します。
2. cmx_delete_system コマンドを使用して，手順 1.で指定した Web システム内のサービスユニットを削除します。

4.9.4　システムの運用手順の検討

ここでは，「システム運用手順書」に記載するシステムの運用手順の例について説明します。「システム運用手順書」を作成するときは，ここで説明している手順を参考にしてください。

セキュアなシステムの運用には，Smart Composer 機能のコマンド，および snapshotlog コマンドを使用します。ここで説明している以外の操作を作業手順書に記載する場合も，監査ログが出力されるコマンドを使用する方法を記載してください。監査ログが出力されるコマンドについては，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「6.6　監査ログを出力するコマンド・操作一覧」を参照してください。

なお，この項で説明する手順のうち，Web システムの起動の作業の一部，および Web システムのメンテナンス以外は，システム運用者が実行します。Web システムのメンテナンスは，システム運用者に依頼されたシステム管理者が実行します。

(1) Web システムの起動

システム運用者が，運用管理サーバの管理者端末で Web システムを起動します。ただし，リソースアダプタおよび J2EE アプリケーションの起動は，システム管理者が，サーバ管理コマンドを使用して実行します。Web システムを起動する手順を次に示します。

1. システム運用者が，cmx_start_target コマンドを使用して，Web システム内のサービスユニットを準備状態にします。
2. システム運用者が，システム管理者にサービスの起動を依頼します。
3. システム管理者が，cjstartrar コマンドを使用して，リソースアダプタを開始します。
4. システム管理者が，cjstartapp コマンドを使用して，J2EE アプリケーションを開始します。
5. システム運用者が，cmx_start_target コマンドを使用して，Web システム内のサービスユニットを稼働状態にします。

(2) Web システムの停止

システム運用者が，運用管理サーバの管理者端末で Smart Composer 機能のコマンドを使用して，Web システムを停止します。Web システムを停止する手順を次に示します。

1. cmx_stop_target コマンドを使用して，Web システム内のサービスユニットを停止状態にします。

(3) エンドユーザの管理

システム運用者が，「システム運用手順書」に従って，エンドユーザのアクセス権やユーザ ID などを管理します。次の作業を実施します。

- エンドユーザの登録，削除
- エンドユーザの権限の変更
- エンドユーザのパスワード変更

なお，「システム運用手順書」では，これらの手順の詳細を規定してください。

(4) エンドユーザへの通知

システム運用者が，「(3) エンドユーザの管理」で登録したエンドユーザのユーザ ID とパスワードをエンドユーザに通知します。ユーザ ID とパスワードをエンドユーザに通知する手順を次に示します。

1. 「システム運用手順書」に従って登録したエンドユーザのユーザ ID とパスワードを使用して，サービスを利用できることを確認します。

2. サービス利用時に，監査ログが出力されることを確認します。

3. 手順 1.および手順 2.の確認が終わったら，システム運用者がエンドユーザにユーザ ID とパスワードを安全な手段で通知します。

(5) Web システムのメンテナンス

必要に応じて Web システムをメンテナンスします。Web システムのメンテナンスは，システム運用者に依頼されたシステム管理者が実施します。Web システムをメンテナンスする手順を次に示します。

1. システム運用者が，cmx_stop_target コマンドを使用して，メンテナンスするサービスユニットを閉塞，または停止します。

2. アプリケーションサーバの修正パッチを適用する場合，システム運用者がアプリケーションサーバ関連のプログラムを停止します。修正パッチを適用しない場合は，手順 3.に進みます。

3. システム管理者にシステムのメンテナンスを依頼します。

4. システム管理者が，システム管理者用のユーザ ID で，OS にログインします。

5. システム管理者が OS のサービスパック，セキュリティパッチ，アプリケーションサーバの修正パッチなどを適用します。この作業をシステム管理者が実施しているとき，システム運用者はその場に立ち会う必要があります。

6. アプリケーションサーバの修正パッチを適用した場合，システム運用者がアプリケーションサーバ関連のプログラムを再開します。

7. システム運用者が，cmx_start_target コマンドを使用して，サービスユニットを再開します。

(6) システムの障害対応

障害発生時に，システム運用者が障害に対応します。システムの障害に対応する手順を次に示します。

1. snapshotlog コマンドを使用して，アプリケーションサーバのログを収集します。

2. 必要に応じて，snapshotlog コマンドで収集できないログを個別に収集します。

3. 収集したログを保守員に送付して，調査を依頼します。

4. 調査の結果に基づいてシステムをメンテナンスします。

4.10　システムの監査方法の確認

この節では，システムの監査方法について説明します。

システムの監査では，監査者がそれぞれの作業手順書と監査ログに出力されている操作の記録を照合して，正しい作業者が正しい手順で作業を実行しているかどうかを調査します。

4.10.1　監査ログの入手

監査ログを入手する手順について次に示します。

1. 監査者が，「入退室手順書」に従って外部から物理的に隔離されたサーバエリアに入室します。
2. 監査者が，監査ログサーバにログインして，各種サーバが稼働しているマシンから監査ログを入手します。

4.10.2　監査ログの調査

監査ログの調査では，次の点を確認します。

1. 信頼できるシステム管理者が正しい方法でシステムを構築しているか
   監査ログに出力された時刻，操作者，事象内容，および結果と，「システム構築手順書」の内容に相違がないことを確認します。
2. 正しい方法で構築されたシステムが，正しく運用，および利用されているか
   監査ログに出力された時刻，操作者，事象内容，および結果と，「システム運用手順書」および「エンドユーザ操作手順書」に相違がないことを確認します。

監査ログの調査方法，および監査ログに出力されるメッセージの詳細については，次のマニュアルを参照してください。

- 監査ログの調査方法
  マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「6.3　監査ログとは」を参照してください。
- 監査ログに出力されるメッセージの詳細
  マニュアル「アプリケーションサーバ メッセージ(監査者用)」を参照してください。

4.11　外部ネットワークを使用するシステムでのセキュリティの検討

この節では，外部ネットワークを使用するシステムで想定されるセキュリティ上の脅威とその対策について説明します。

4.11.1　外部ネットワークを使用するシステムで想定されるセキュリティ上の脅威

ここでは，外部ネットワークを使用するシステムで想定されるセキュリティ上の脅威について説明します。

(1)　想定するセキュリティ上の脅威

ネットワークを使用するシステムでは，セキュリティ対策が十分でないと，アプリケーションを不正に実行されたり，通信内容やバックエンドのデータベースで管理しているデータが漏洩または改ざんされたりするおそれがあります。これらを防ぐためには，セキュリティ上の脅威を認識して，それらに十分に対策しておく必要があります。

ここでは，次のようなセキュリティ上の脅威を想定します。

- システム外からシステムへの第三者の不正侵入
- アプリケーションの扱うデータの第三者への漏洩
- アプリケーションの通信内容の第三者への漏洩
- アプリケーションの通信内容の第三者による改ざん
- システム利用者の許可された権限の範囲を超えた操作や情報の入手

なお，ここでは，システム外部からの脅威を想定した対策について検討します。システム内部の脅威については，ここでは対象にしません。

(2)　考えられる対策

想定したセキュリティ上の脅威に対しては，それぞれ，次の表に示すような対策が考えられます。それぞれの具体的な対策方法については，参照先の説明を参照してください。

表 4-6　セキュリティ上の脅威に対して考えられる対策

脅威	対策	参照先
システム外からシステムへの第三者の不正侵入	ファイアウォールと侵入検知システムを配置する	4.11.2
アプリケーションの扱うデータの第三者への漏洩		
アプリケーションの通信内容の第三者への漏洩	通信内容を暗号化する	4.11.3※
アプリケーションの通信内容の第三者による改ざん		

脅威	対策	参照先
システム利用者の許可された権限の範囲を超えた操作や情報の入手	アプリケーションでユーザを認証する	4.11.4

注※　通信内容の暗号化には，HTTPS を使用します。参照先では，HTTPS を使用する場合に，SSL アクセラレータを使用して暗号通信を処理する方法について説明します。

4.11.2　ファイアウォールと侵入検知システムを配置する

ここでは，ファイアウォールと侵入検知システムを適切に配置，設定することでシステムのセキュリティを向上させる方法について説明します。

(1)　ファイアウォールと侵入検知システムを配置する目的

ファイアウォールは，外部のネットワークと内部のネットワーク間のアクセスを制御します。あらかじめアクセスを許可するクライアントや通信を特定し，決められたルールに従って通信を許可したり破棄したりすることで，外部ネットワークからの不正なアクセスを防止します。このため，ファイアウォールを使用する場合は，通信を許可するポートや IP アドレスを明確にして，設定しておく必要があります。

侵入検知システム（IDS）は，通信回線を監視して，通信パターンによって不正なアクセスを判断します。

ファイアウォールと侵入検知システムを適切な個所に配置，設定することで，次のようなセキュリティ上の脅威からシステムを守れます。

- システム外からシステムへの第三者の不正侵入
- アプリケーションの扱うデータの第三者への漏洩

ここでは，次の表に示すシステム構成ごとのファイアウォールと侵入検知システムの配置個所，および設定するときに留意することについて説明します。

表 4–7　ファイアウォールと侵入検知システムの配置を検討するためのシステム構成の分類

システム構成	説明
基本的な Web クライアント構成	1 台のアプリケーションサーバで構成されるシステム構成です。クライアントは Web ブラウザです。
基本的な EJB クライアント構成	1 台のアプリケーションサーバで構成されるシステム構成です。クライアントは EJB クライアントアプリケーションです。
サーバの層ごとにファイアウォールで区切る構成（アプリケーション集中型）	複数のアプリケーションサーバで構成されるシステム構成で，サーバの層ごとにファイアウォールで区切る構成です。アプリケーションはすべて同じ層のアプリケーションサーバ上で動作します。
サーバの層ごとにファイアウォールで区切る構成（アプリケーション分散型）	複数のアプリケーションサーバで構成されるシステム構成で，サーバの層ごとにファイアウォールで区切る構成です。アプリケーションは異なる複数の層のアプリケーションサーバ上で動作します。

なお，インターネットと接続するシステムの場合は，外部ネットワークから直接内部ネットワークのアプリケーションサーバにアクセスされないように，DMZ を確保して，リバースプロキシを使用した構成を検討することをお勧めします。

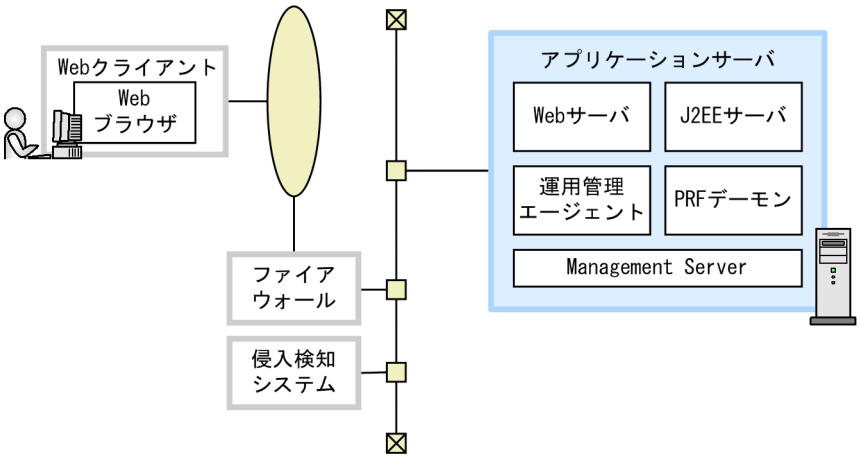
(2)　基本的な Web クライアント構成の場合

ここでは，1 台のアプリケーションサーバで構成される基本的な Web クライアント構成の場合のファイアウォールと侵入検知システムの配置個所について説明します。

ファイアウォールは，ネットワークから見てアプリケーションサーバの手前に配置します。この構成の場合，ネットワーク上の Web クライアントは，ファイアウォール経由でだけアプリケーションサーバにアクセスできます。

基本的な Web クライアント構成の場合のファイアウォールと侵入検知システムの配置例を次の図に示します。

図 4-4　基本的な Web クライアント構成でのファイアウォールと侵入検知システムの配置例



(a)　アプリケーションサーバの設定

アプリケーションサーバでは，次の項目を設定します。

- J2EE サーバの管理用通信ポートのアドレス指定
  J2EE サーバの管理用通信ポートにアクセスできるアドレスを指定します。
- Management Server および運用管理エージェントのアドレス指定
  Management Server および運用管理エージェントにアクセスできるアドレスを指定します。

(b)　ファイアウォールの設定

外部ネットワークとアプリケーションサーバ内の Web サーバ（HTTP Server）のアクセスについて制御するために，次の項目を設定します。

- 外部ネットワークからの Web サーバに対してのアクセス許可

ファイアウォールの外部のネットワークとアプリケーションサーバ間の通信では，公開ポートであるHTTP/80 や HTTPS/443 などだけを許可するようにします。ただし，システム構成によっては，DNSなど，そのほかの通信を必要に応じて許可してください。

- アクセスを許可する Web クライアントの IP アドレスの限定（任意）

  ファイアウォールの機能を使用してアクセスを許可する Web クライアントの IP アドレスを限定することで，そのシステムの利用者を特定できます。この場合は，ファイアウォールに通信を許可する IP アドレスを指定してください。

- Management Server および運用管理エージェントの通信ポートの指定

  Management Server および運用管理エージェントの通信ポートについては，ファイアウォールの外部からはアクセスできないように通信を遮断してください。これらの通信ポートにアクセスできると，運用管理者以外の外部のユーザからアプリケーションサーバに対して不正な運用操作が実行されるおそれがあります。

(c) 侵入検知システムの設定

外部ネットワークとアプリケーションサーバ内の Web サーバ（HTTP Server）の公開ポートの通信内容を監視するために，次の項目を設定します。

- 通信内容の監視

  通信内容に既知の攻撃パターンや，攻撃と疑われるようなパターンが含まれている場合は運用管理者などに警告を通知するように設定します。侵入検知システムとファイアウォールの連携機能を利用して，疑わしい通信は自動的に遮断するように設定することもできます。

- SSL コネクションの確立部分への攻撃の監視

  HTTPS によるアクセスは，通信内容が暗号化されているため，基本的に通信内容を監視できません。この場合は，HTTPS に関する既知の攻撃パターンとして，SSL コネクションの確立部分への攻撃などを監視対象としてください。

- 公開ポート以外のポートへの通信の監視

  アプリケーションサーバの公開ポート以外のポートへの通信が外部ネットワークから送信されている場合は，設定ミスなどによってファイアウォールが突破されているおそれがあります。この場合，警告を通知するように設定することをお勧めします。

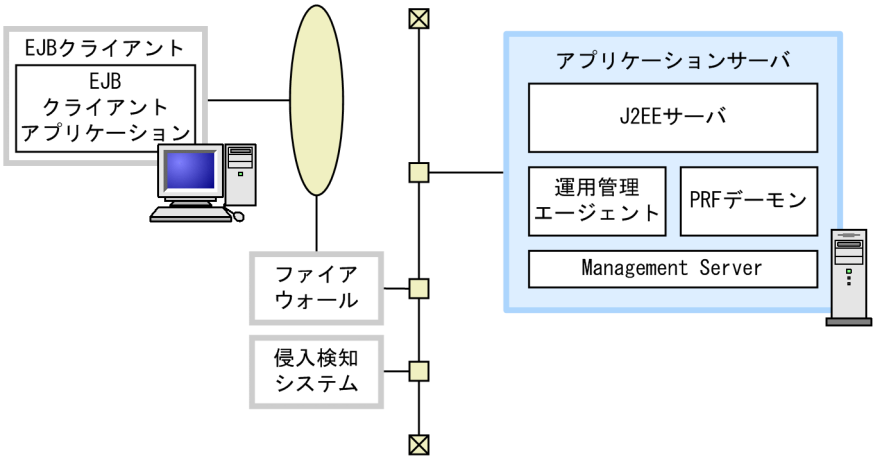
(3) 基本的な EJB クライアント構成の場合

ここでは，1 台のアプリケーションサーバで構成される基本的な EJB クライアント構成の場合のファイアウォールと侵入検知システムの配置個所について説明します。

ファイアウォールは，ネットワークから見てアプリケーションサーバの手前に配置します。この構成の場合，ネットワーク上の EJB クライアントは，ファイアウォール経由でだけアプリケーションサーバにアクセスできます。

基本的な EJB クライアント構成の場合のファイアウォールと侵入検知システムの配置例を次の図に示します。

図 4-5　基本的な EJB クライアント構成でのファイアウォールと侵入検知システムの配置



(a)　アプリケーションサーバの設定

アプリケーションサーバでは，次の項目を設定します。

- **J2EE サーバの管理用通信ポートのアドレス指定**

  J2EE サーバの管理用通信ポートにアクセスできるアドレスを指定します。

- **Management Server および運用管理エージェントのアドレス指定**

  Management Server および運用管理エージェントにアクセスできるアドレスを指定します。

- **EJB クライアントからアクセスされるポート番号の固定**

  EJB クライアントからアプリケーションサーバを利用するために，EJB クライアントから次のポート番号に通信できるように設定してください。

  - CORBA ネーミングサービス

    ポート番号は，通常固定されています（デフォルトのポート番号：900）。

  - EJB コンテナ

    ポート番号が固定されていないため，EJB コンテナが利用するポート番号を明示的に指定（固定）する必要があります。指定個所については，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.15　アプリケーションサーバのプロセスが使用する TCP/UDP のポート番号」を参照してください。

- **Management Server および運用管理エージェントの通信ポートの指定**

  Management Server および運用管理エージェントの通信ポートの指定については，ファイアウォールの外部からはアクセスできないように，公開ポートを使用しないことを推奨します。これらの通信ポートにアクセスできると，運用管理者以外の外部のユーザからアプリケーションサーバに対して不正な運用操作が実行されるおそれがあります。

(b)　ファイアウォールの設定

外部ネットワークとアプリケーションサーバ間のアクセスについて制御するために，次の項目を設定します。

- **外部ネットワークからアプリケーションサーバに対してのアクセス許可**

ファイアウォールの外部のネットワークとアプリケーションサーバ間の通信では，公開ポートであるCORBA ネーミングサービスおよび EJB コンテナの固定ポートなどだけを許可するようにします。ただし，システム構成によっては，DNS など，そのほかの通信を必要に応じて許可してください。

- アクセスを許可するクライアントの IP アドレスの限定（任意）

  ファイアウォールの機能を使用してアクセスを許可するクライアントの IP アドレスを限定することで，そのシステムの利用者を特定できます。この場合は，ファイアウォールに通信を許可する IP アドレスを指定してください。

- Management Server および運用管理エージェントの通信ポートの指定

  Management Server および運用管理エージェントの通信ポートについては，ファイアウォールの外部からはアクセスできないように通信を遮断してください。これらの通信ポートにアクセスできると，運用管理者以外の外部のユーザからアプリケーションサーバに対して不正な運用操作が実行されるおそれがあります。

(c) 侵入検知システムの設定

外部ネットワークとアプリケーションサーバの公開ポートの通信内容を監視するために，次の項目を設定します。

- 通信内容の監視

  通信内容に既知の攻撃パターンや，攻撃と疑われるようなパターンが含まれている場合は運用管理者などに警告を通知するように設定します。侵入検知システムとファイアウォールの連携機能を利用して，疑わしい通信は自動的に遮断するように設定することもできます。

- SSL コネクションの確立部分への攻撃の監視

  HTTPS によるアクセスは，通信内容が暗号化されているため，基本的に通信内容を監視できません。この場合は，HTTPS に関する既知の攻撃パターンとして，SSL コネクションの確立部分への攻撃などを監視対象としてください。

- 公開ポート以外のポートへの通信の監視

  アプリケーションサーバの公開ポート以外のポートへの通信が外部ネットワークから送信されている場合は，設定ミスなどによってファイアウォールが突破されているおそれがあります。この場合，警告を通知するように設定することをお勧めします。

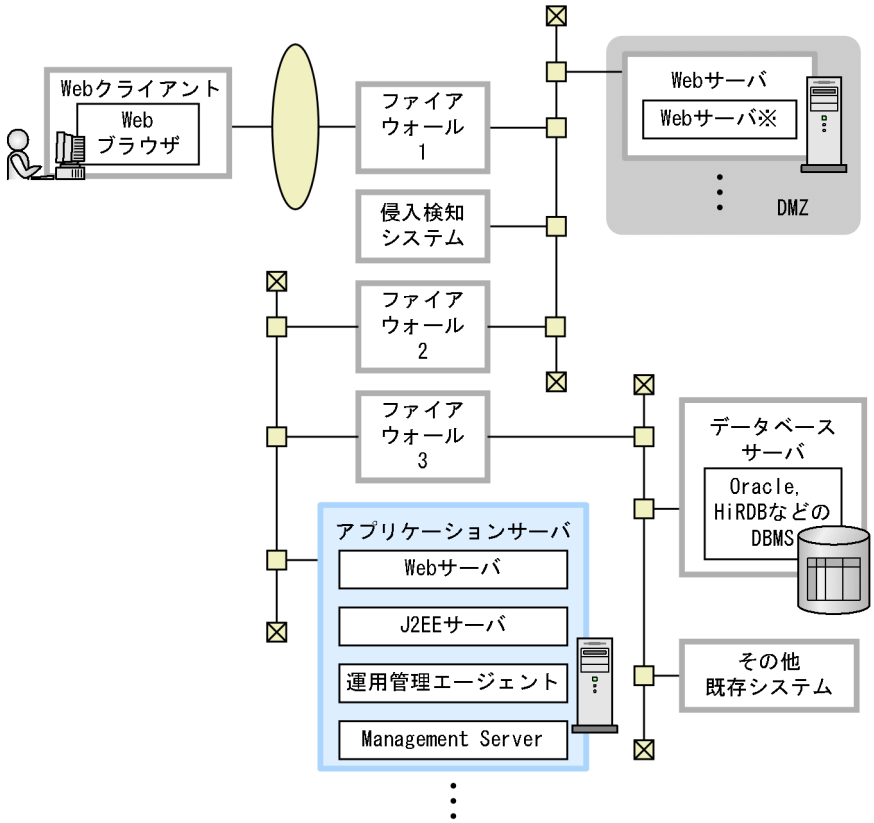
(4) サーバの層ごとにファイアウォールで区切る構成（アプリケーション集中型）

システムの規模によっては，一つのシステムが複数のアプリケーションサーバおよびそのほかのサーバで構成されることがあります。このような構成では，それぞれの層でセキュリティを確保する必要があります。

ここでは，Web サーバ，アプリケーションサーバおよびデータベースサーバを多層化した場合に，アプリケーションはすべて同じ層のアプリケーションサーバで動作させる構成について説明します。この構成を，**アプリケーション集中型の構成**といいます。

アプリケーション集中型の構成の場合のファイアウォールと侵入検知システムの配置例を次の図に示します。この構成では，サーバの層ごとに 1 台ずつ，合計 3 台のファイアウォールを配置しています。また，DMZ には，リバースプロキシモジュールを組み込んだ Web サーバ（リバースプロキシサーバ）を配置しています。

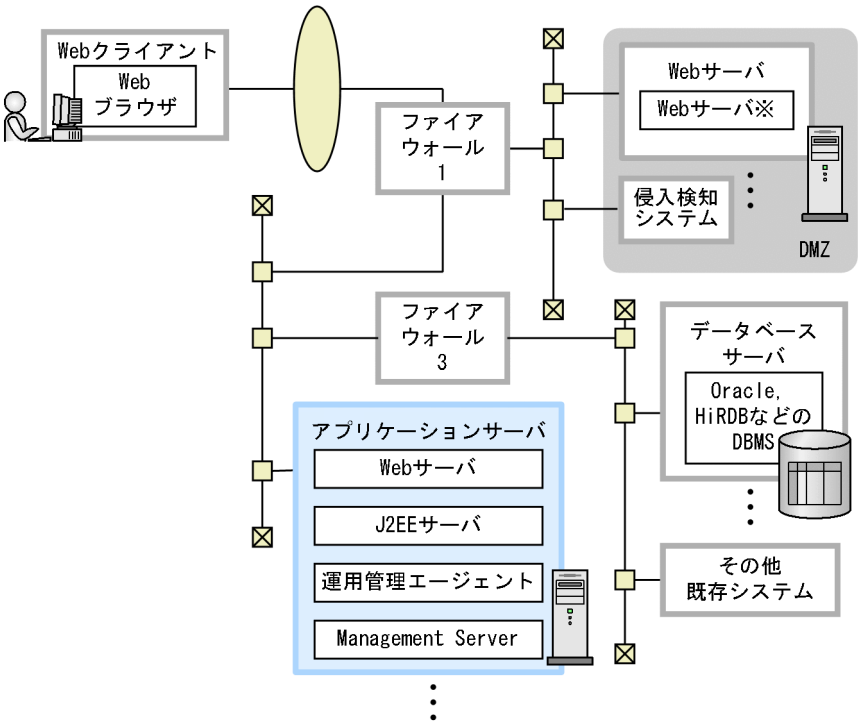
図 4-6　アプリケーション集中型のファイアウォールと侵入検知システムの配置



注※　リバースプロキシモジュールを組み込んだWebサーバです。DMZに配置します。

また，コストなどの要因からファイアウォール数を削減したい場合は，次の図に示すような構成にできます。この例では，ファイアウォール 1 とファイアウォール 2 で実行するアクセス制御をファイアウォール 1 に集約して，ファイアウォール 2 を削減します。

図 4-7　ファイアウォールを削減した構成



注※　リバースプロキシモジュールを組み込んだWebサーバです。DMZに配置します。

この構成の場合は，ファイアウォール 2 で設定する内容を，ファイアウォール 1 にまとめて設定してください。

(a)　アプリケーションサーバの設定

アプリケーションサーバでは，次の項目を設定します。

- **J2EE サーバの管理用通信ポートのアドレス指定**
  J2EE サーバの管理用通信ポートにアクセスできるアドレスを指定します。
- **Management Server および運用管理エージェントのアドレス指定**
  Management Server および運用管理エージェントにアクセスできるアドレスを指定します。

(b)　ファイアウォールの設定

この構成では，次に示す三つのファイアウォールを使用しています。

- ファイアウォール 1
  外部のネットワークと DMZ の Web サーバ（リバースプロキシサーバ）間のアクセスを制御します。
- ファイアウォール 2
  DMZ の Web サーバ（リバースプロキシサーバ）と内部のネットワークにあるアプリケーションサーバ間のアクセスを制御します。
- ファイアウォール 3
  アプリケーションサーバとデータベースサーバ間のアクセスを制御します。

それぞれのファイアウォールに設定する内容について説明します。

- **ファイアウォール 1 の設定**

  外部のネットワークと DMZ の Web サーバ（リバースプロキシサーバ）間のアクセスを制御するためのファイアウォールです。次の項目を設定します。

  - **外部ネットワークから Web サーバ（リバースプロキシサーバ）に対してのアクセス許可**

    ファイアウォール 1 よりも外部のネットワークからアプリケーションサーバ内の Web サーバに対する通信は，公開ポートへの通信だけを許可します。HTTP/80 や HTTPS/443 などが該当します。ただし，システム構成によっては，DNS など，そのほかの通信を必要に応じて許可してください。

  - **アクセスを許可する Web クライアントの IP アドレスの限定（任意）**

    ファイアウォールの機能を使用してアクセスを許可する Web クライアントの IP アドレスを限定することで，そのシステムの利用者を特定できます。この場合は，ファイアウォール 1 に通信を許可する IP アドレスを指定してください。

- **ファイアウォール 2 の設定**

  Web サーバとアプリケーションサーバ間のアクセスを制御するためのファイアウォールです。次の項目を設定します。

  - **DMZ の Web サーバ（リバースプロキシサーバ）から内部ネットワークのアプリケーションサーバ内の Web サーバに対してのアクセス許可**

    ファイアウォール 2 よりも外部のネットワーク（DMZ）からアプリケーションサーバ内の Web サーバに対する通信は，公開ポートへの通信だけを許可します。HTTP/80 や HTTPS/443 などが該当します。ただし，システム構成によっては，DNS など，そのほかの通信を必要に応じて許可してください。

  - **アクセスを許可する Web クライアントの IP アドレスの限定（任意）**

  ファイアウォールの機能を使用してアクセスを許可する Web クライアントの IP アドレスを限定することで，そのシステムの利用者を特定できます。この場合は，リバースプロキシサーバの IP アドレスを指定してください。

  このほかの通信の設定は，システム構成によって，必要に応じて許可してください。DNS などの通信の許可が必要な場合があります。

  参考

  Web サーバと J2EE サーバが動作するアプリケーションサーバの間にファイアウォールを配置した場合には，次の設定が必要です。

  - **Web サーバからアプリケーションサーバに対してのアクセス許可**

    J2EE サーバの Web サーバ通信用ポート（NIO HTTP サーバのリクエスト受付ポート。デフォルトのポート番号：8008）への通信の許可を設定します。

- **ファイアウォール 3 の設定**

アプリケーションサーバとデータベース間のアクセスを制御するためのファイアウォールです。このファイアウォールが，システムで最も重要な情報を守るための最終防衛ラインになります。

次の項目を設定します。

- アプリケーションサーバからデータベースサーバに対してのアクセス許可

  アプリケーションサーバからデータベースサーバに対する通信は，データベースサーバの通信用ポートだけを許可するように設定します。データベースサーバの通信用ポートは，使用するデータベースの設定に従ってください。データベースサーバからアプリケーションサーバへのコネクション確立が必要な場合があるので注意してください。

このほかの通信の設定は，システム構成によって，必要に応じて許可してください。DNS などの通信の許可が必要な場合があります。

(c) 侵入検知システムの設定

外部ネットワークとアプリケーションサーバ内の Web サーバの公開ポートの通信内容を監視するために，次の項目を設定します。

- 通信内容の監視

  通信内容に既知の攻撃パターンや，攻撃と疑われるようなパターンが含まれている場合は運用管理者などに警告を通知するように設定します。侵入検知システムとファイアウォールの連携機能を利用して，疑わしい通信は自動的に遮断するように設定することもできます。

- SSL コネクションの確立部分への攻撃の監視

  HTTPS によるアクセスは，通信内容が暗号化されているため，基本的に通信内容を監視できません。この場合は，HTTPS に関する既知の攻撃パターンとして，SSL コネクションの確立部分への攻撃などを監視対象としてください。

- 公開ポート以外のポートへの通信の監視

  アプリケーションサーバの公開ポート以外のポートへの通信が外部ネットワークから送信されている場合は，設定ミスなどによってファイアウォールが突破されているおそれがあります。この場合，警告を通知するように設定することをお勧めします。

(5) サーバの層ごとにファイアウォールで区切る構成（アプリケーション分散型）

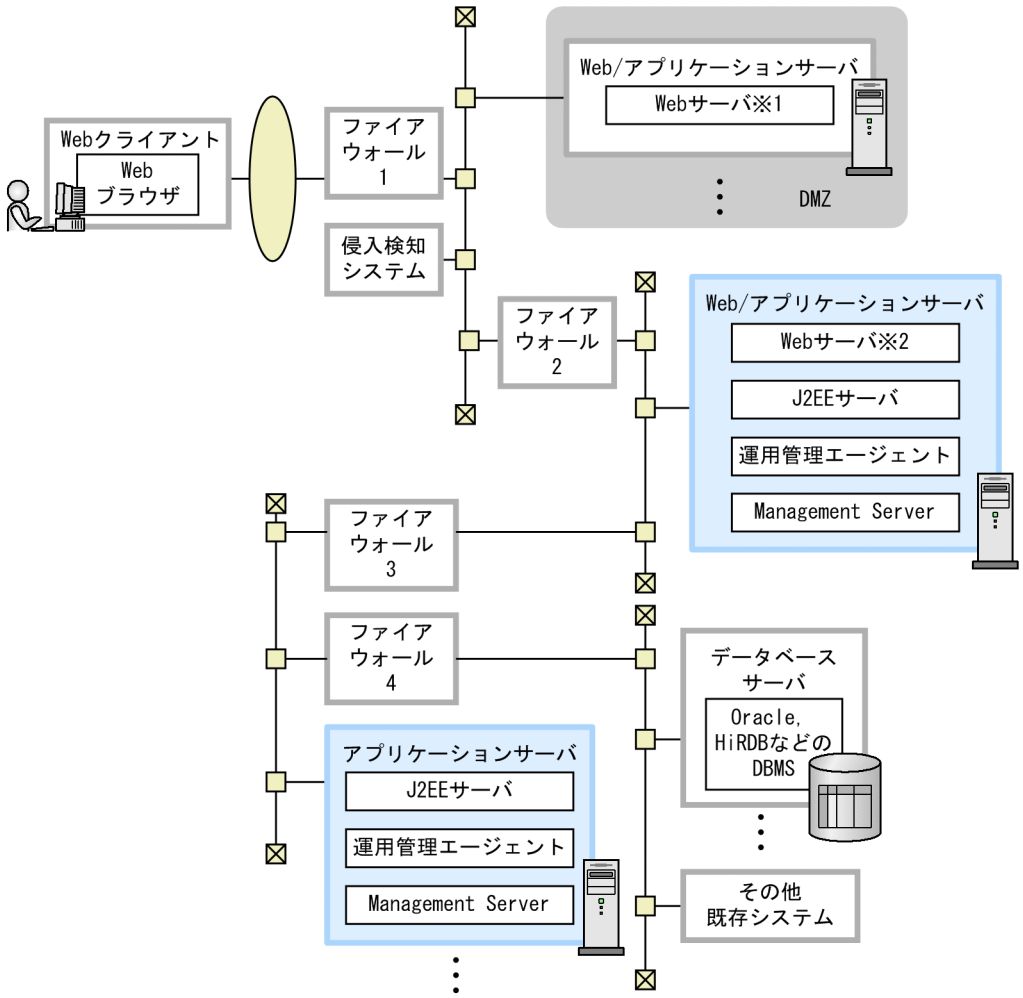
ここでは，Web サーバ，アプリケーションサーバおよびデータベースサーバを多層化した場合に，アプリケーションを複数の層のアプリケーションサーバで動作させるときの構成について説明します。この構成を，**アプリケーション分散型の構成**といいます。

アプリケーション分散型の構成の場合のファイアウォールと侵入検知システムの配置例を次の図に示します。この例では，Web サーバに該当するマシンがアプリケーションサーバを兼ねているため，Web アプリケーションが Web サーバと同じ層で動作します。また，Enterprise Bean は，Web サーバとは別のマシン上に構築されたアプリケーションサーバ上で動作します。

なお，運用管理は，各ホストに配置した Management Server によって実行します。このため，管理ホストは層ごとに配置しています。

この構成では，DMZ のフロントに 1 台，およびサーバの層ごとに 1 台ずつ，合計 4 台のファイアウォールを配置しています。また，DMZ には，リバースプロキシモジュールを組み込んだ Web サーバ（リバースプロキシサーバ）を配置しています。

図 4-8　アプリケーション分散型のファイアウォールと侵入検知システムの配置



注※1　リバースプロキシモジュールを組み込んだWebサーバです。DMZに配置します。
注※2　リダイレクタモジュールを組み込んだWebサーバです。
　　アプリケーションサーバと同じマシンに配置します。

(a)　Web／アプリケーションサーバでの設定

Web サーバを兼ねたアプリケーションサーバマシン（Web／アプリケーションサーバ）では，次の項目を設定します。なお，このアプリケーションサーバマシンでは，Web アプリケーションが動作します。

- **J2EE サーバの管理用通信ポートのアドレス指定**

  J2EE サーバの管理用通信ポートにアクセスできるアドレスを指定します。

- **Management Server および運用管理エージェントのアドレス指定**

  Management Server および運用管理エージェントにアクセスできるアドレスを指定します。

(b) アプリケーションサーバの設定

Enterprise Bean が動作するアプリケーションサーバでは，次の項目を設定します。

- **J2EE サーバの管理用通信ポートのアドレス指定**
  J2EE サーバの管理用通信ポートにアクセスできるアドレスを指定します。
- **Management Server および運用管理エージェントのアドレス指定**
  Management Server および運用管理エージェントにアクセスできるアドレスを指定します。
- **EJB コンテナが利用するポート番号の固定**
  EJB コンテナが利用するポート番号を明示的に指定（固定）する必要があります。
  指定個所については，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.15　アプリケーションサーバのプロセスが使用する TCP/UDP のポート番号」を参照してください。

(c) ファイアウォールの設定

この構成では，次の 4 台のファイアウォールを使用しています。

- ファイアウォール 1
  外部のネットワークと DMZ の Web サーバ（リバースプロキシサーバ）間のアクセスを制御します。
- ファイアウォール 2
  DMZ の Web サーバ（リバースプロキシサーバ）と内部のネットワーク上の Web／アプリケーションサーバ間のアクセスを制御します。
- ファイアウォール 3
  Web／アプリケーションサーバとアプリケーションサーバ間のアクセスを制御します。
- ファイアウォール 4
  アプリケーションサーバとデータベースサーバ間のアクセスを制御します。

それぞれのファイアウォールに設定する内容について説明します。

- **ファイアウォール 1 の設定**
  外部のネットワークと DMZ の Web サーバ（リバースプロキシサーバ）間のアクセスを制御するためのファイアウォールです。次の項目を設定します。
  - **外部ネットワークから Web サーバ（リバースプロキシサーバ）に対してのアクセス許可**
    ファイアウォール 1 よりも外部のネットワークからアプリケーションサーバ内の Web サーバに対する通信は，公開ポートへの通信だけを許可します。HTTP/80 や HTTPS/443 などが該当します。ただし，システム構成によっては，DNS など，そのほかの通信を必要に応じて許可してください。
  - **アクセスを許可する Web クライアントの IP アドレスの限定（任意）**
    ファイアウォールの機能を使用してアクセスを許可する Web クライアントの IP アドレスを限定することで，そのシステムの利用者を特定できます。この場合は，ファイアウォール 1 に通信を許可する IP アドレスを指定してください。

- ファイアウォール 2 の設定

  外部のネットワークと内部のネットワーク上の Web／アプリケーションサーバ間のアクセスを制御するためのファイアウォールです。次の項目を設定します。

  - DMZ の Web サーバ（リバースプロキシサーバ）からアプリケーションサーバ内の Web サーバに対してのアクセス許可

    ファイアウォール 1 よりも外部のネットワークから，アプリケーションサーバ内の Web サーバに対する通信は，公開ポートへの通信だけを許可します。HTTP/80 や HTTPS/443 などが該当します。ただし，システム構成によっては，DNS など，そのほかの通信を必要に応じて許可してください。

  - アクセスを許可する Web クライアントの IP アドレスの限定（任意）

    ファイアウォールの機能を使用してアクセスを許可する Web クライアントの IP アドレスを限定することで，そのシステムの利用者を特定できます。この場合は，リバースプロキシサーバの IP アドレスを指定してください。

- ファイアウォール 3 の設定

  Web／アプリケーションサーバとアプリケーションサーバ間のアクセスを制御するためのファイアウォールです。次の項目を設定します。

  - Web／アプリケーションサーバからアプリケーションサーバに対してのアクセス許可

    Web／アプリケーションサーバからアプリケーションサーバ上の J2EE サーバを利用するために，次のポート番号に通信できるように設定してください。

    ・CORBA ネーミングサービス

    ポート番号は，通常固定されています（デフォルトのポート番号：900）。

    ・EJB コンテナ

    ポート番号が固定されていないため，EJB コンテナが利用するポート番号を明示的に指定（固定）する必要があります。

    指定個所については，マニュアル「アプリケーションサーバ システム設計ガイド」の「3.15　アプリケーションサーバのプロセスが使用する TCP/UDP のポート番号」を参照してください。

  - トランザクション関連の双方向アクセスの許可（グローバルトランザクションでトランザクションコンテキストプロパゲーションを使用する場合）

    Web／アプリケーションサーバとアプリケーションサーバの間で，グローバルトランザクションでトランザクションコンテキストプロパゲーションを使用する場合は，両方のアプリケーションサーバの次に示すポートが双方向で通信できるように設定します。

    ・J2EE サーバのトランザクションリカバリ用通信ポート（デフォルトのポート番号：20302）

    ・スマートエージェントの通信ポート（デフォルトのポート番号：14000）

  - そのほかの設定（任意）

    システム構成によっては，DNS など，そのほかの通信を必要に応じて許可してください。

- ファイアウォール 4 の設定

  アプリケーションサーバとデータベース間のアクセスを制御するためのファイアウォールです。このファイアウォールが，システムで最も重要な情報を守るための最終防衛ラインになります。

次の項目を設定します。

- アプリケーションサーバからデータベースサーバに対してのアクセス許可

  アプリケーションサーバからデータベースサーバに対する通信は，データベースサーバの通信用ポートだけを許可するように設定します。データベースサーバの通信用ポートは，使用するデータベースの設定に従ってください。データベースサーバからアプリケーションサーバへのコネクション確立が必要な場合があるので注意してください。また，システム構成によっては，DNS など，そのほかの通信を必要に応じて許可してください。

(d) 侵入検知システムの設定

外部ネットワークとアプリケーションサーバ内にある Web サーバの公開ポートの通信内容を監視するために，次の項目を設定します。

- 通信内容の監視

  通信内容に既知の攻撃パターンや，攻撃と疑われるようなパターンが含まれている場合は運用管理者などに警告を通知するように設定します。侵入検知システムとファイアウォールの連携機能を利用して，自動的に疑わしい通信は遮断するように設定することもできます。

- SSL コネクションの確立部分への攻撃の監視

  HTTPS によるアクセスは，通信内容が暗号化されているため，基本的に通信内容を監視できません。この場合は，HTTPS に関する既知の攻撃パターンとして，SSL コネクションの確立部分への攻撃などを監視対象としてください。

- 公開ポート以外のポートへの通信の監視

  Web／アプリケーションサーバの公開ポート以外のポートへの通信が外部ネットワークから送信されている場合は，設定ミスなどによってファイアウォールが突破されているおそれがあります。この場合，警告を通知するように設定することをお勧めします。

4.11.3 SSL アクセラレータを使用して暗号通信を処理する

ここでは，SSL アクセラレータを使用して暗号通信を処理する方法について説明します。

(1) SSL アクセラレータを使用する目的

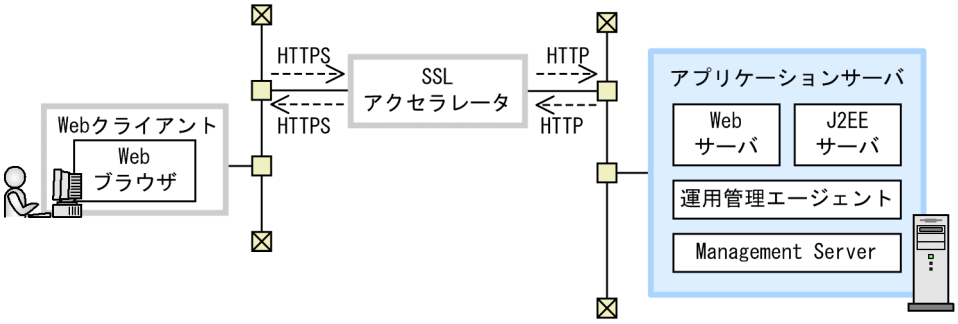
セキュリティ上の脅威のうち，アプリケーションの通信内容の第三者への漏洩または改ざんを防ぐ方法の一つに，通信内容の暗号化があります。暗号化手段としては，通信に HTTPS を使用する方法があります。ただし，HTTPS の基盤となる TLS／SSL による通信は大変負荷が高い処理です。

SSL アクセラレータは，このような場合に，Web サーバやアプリケーションサーバに負荷を掛けないで，HTTPS によって暗号化された通信内容の処理を実現するための専用ハードウェアです。SSL アクセラレータを適切に配置することで，Web サーバやアプリケーションサーバに負荷を掛けることなく，暗号化された内容の通信を高速化できます。

(2) SSL アクセラレータの配置

SSL アクセラレータを適用した構成の例を次の図に示します。

図 4-9 SSL アクセラレータを適用した構成



Web クライアントから HTTPS によって送信された通信内容は，SSL アクセラレータによって復号化され，HTTP によって Web サーバまたはアプリケーションサーバに送信されます。また，Web サーバまたはアプリケーションサーバから HTTP によって送信された通信内容は，SSL アクセラレータによって暗号化されて，Web クライアントに送信されます。

SSL アクセラレータを配置する場合は，次の点に留意してください。

- SSL アクセラレータは，ファイアウォールとの併用もできます。併用する場合は，SSL アクセラレータを Web サーバまたはアプリケーションサーバと一体として扱ってください。

4.11.4 アプリケーションでユーザを認証する

ここでは，Web クライアント構成の場合に，セキュリティ確保のためにアプリケーションで利用できる認証機能について説明します。

(1) アプリケーションでユーザを認証する目的

アプリケーションを実行するユーザを認証することで，セキュリティ上の脅威のうち，システム利用者によって，許可した権限の範囲を超えて操作されたり情報を入手されたりすることを防げます。

アプリケーションサーバでは，次の 3 種類のプロトコルを使用してユーザ認証によるセキュリティ確保ができます。

- HTTPS（クライアント認証）
- HTTP（Basic 認証）
- HTTP（Form 認証）

これらのプロトコルを目的に応じて使い分けて，セキュリティを確保してください。

(2) アプリケーションによるユーザ認証方法の比較

通信プロトコルごとの，認証実施個所および認証に使用されるエンジンについて，次の表に示します。

表 4-8 通信プロトコルごとの認証実施個所および認証に使用されるエンジン

使用するプロトコル	認証実施個所	認証に使用されるエンジン
HTTPS(クライアント認証)	HTTP Server または Microsoft IIS	SSL
	SSL アクセラレータ	SSL
HTTP(Basic 認証)	HTTP Server	HWS パスワードファイル
		LDAP リポジトリ
	J2EE サーバ（Web コンテナ）	パスワードファイル
HTTP(Form 認証)	J2EE サーバ（Web コンテナ）	パスワードファイル
	J2EE サーバ（統合ユーザ管理）	統合ユーザ管理パスワードファイル
		データベース
		LDAP リポジトリ

それぞれのプロトコルおよび認証に使用されるエンジンには特徴があります。それらの特徴を考慮して，システムの目的に合った認証方法を選択してください。

(a) プロトコルの特徴

アプリケーションサーバのシステムの認証処理に使用できるプロトコルの特徴について，次の表に示します。

表 4-9 プロトコルの特徴

使用するプロトコル	認証インタフェースの自由度	クライアントでの管理の容易さ	ネットワークの安全性
HTTPS (クライアント認証)	Web ブラウザが提供する機能に制約されます。	クライアント証明書が必要です。	暗号化されているため，盗聴されても認証情報は安全です。
HTTP (Basic 認証)	Web ブラウザが提供する機能に制約されます。	一般的なユーザ名称／パスワード形式を使用した認証ができます。	パスワードが平文またはそれと同等の形式で流出します。 このため，通常は，HTTPS（サーバ認証だけ）の暗号機能と併用して使用します。
HTTP (Form 認証)	アプリケーションごとにデザインできます。	一般的なユーザ名称／パスワード形式を使用した認証ができます。	パスワードが平文またはそれと同等の形式で流出します。 このため，通常は，HTTPS（サーバ認証だけ）の暗号機能と併用して使用します。

(b) 認証に使用されるエンジンの特徴

認証に使用されるエンジンの特徴について，次の表に示します。

表 4-10 認証に使用されるエンジンの特徴

エンジンの種類	汎用性	保守性	システム構成への影響	性能影響への影響
パスワードファイル	使用する機能ごとに形式が異なります。	サーバまたはホストごとにユーザ情報が散在します。	特に認証用のプロセスは必要ありません。	認証時にほかのプロセスやホストとの通信が発生しないため，高速です。
データベース	形式によっては，既存のユーザ情報データベースを利用できます。	ユーザ情報の一括管理ができます。	ユーザ情報を格納するデータベースサーバが必要です。	認証時にデータベースアクセスに必要な時間が掛かります。
LDAP リポジトリ	形式によっては，既存のユーザ情報リポジトリを利用できます。	ユーザ情報の一括管理，および分散したユーザ情報の集中管理ができます。	ユーザ情報を格納する LDAP 対応のディレクトリサーバが必要です。	認証時に LDAP ディレクトリサーバへのアクセスに必要な時間が掛かります。

5 統合ユーザ管理による認証

この章では，アプリケーションサーバで構築したシステムのユーザを統合管理するための統合ユーザ管理フレームワークによる認証について説明します。

5.1　この章の構成

統合ユーザ管理フレームワークとは，Java の標準ユーザ認証（JAAS）によるアプリケーションユーザ管理および複数のアプリケーションへのシングルサインオンを実現するフレームワークです。ここでは，統合ユーザ管理フレームワークを使用した統合ユーザ管理について説明します。

この章の構成を次の表に示します。

表 5-1　この章の構成（統合ユーザ管理）

分類	タイトル	参照先
解説	統合ユーザ管理の概要	5.2
	標準ログインモジュールによるユーザ認証の仕組み	5.3
	統合ユーザ管理のセッション管理	5.4
	シングルサインオンの利用方法	5.5
	カスタムログインモジュールの利用	5.6
	ユーザ情報の管理	5.7
	統合ユーザ管理フレームワークが提供する API	5.8
実装	統合ユーザ管理フレームワークによるユーザ認証の実装	5.9
	API を使用したユーザ認証の実装	5.10
	タグライブラリを使用したユーザ認証の実装	5.11
	カスタムログインモジュールを使用したユーザ認証の実装	5.12
設定	統合ユーザ管理機能の設定手順	5.13
	レルム名の決定	5.14
	LDAP ディレクトリサーバの設定	5.15
	ユーザ情報の登録	5.16
	暗号鍵ファイルの作成（シングルサインオンを使用する場合）	5.17
	ユーザ情報の登録（シングルサインオンを使用する場合）	5.18
	コンフィグレーションファイルの作成	5.19
	JavaVM のプロパティの設定	5.20
	ファイルのデプロイ	5.21

注　「運用」および「注意事項」について，この章での説明はありません。

5.2　統合ユーザ管理の概要

統合ユーザ管理は，アプリケーションサーバシステムにログインするユーザについての情報を統合管理する機能です。

統合ユーザ管理を実現するために，Application Server では次の機能を提供しています。

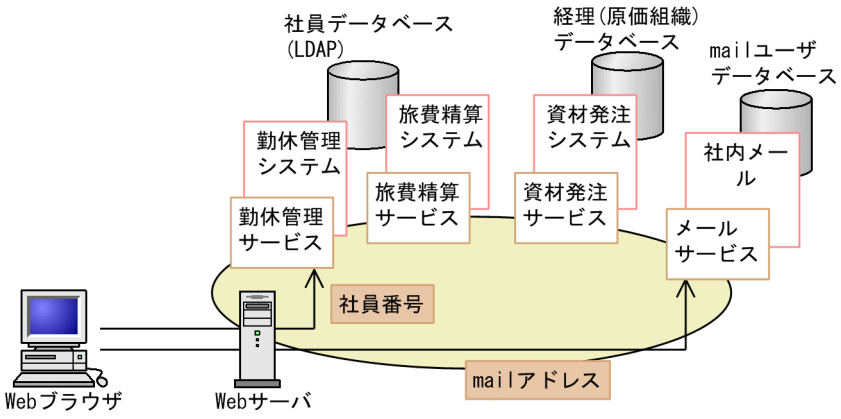
- **統合ユーザ管理フレームワーク**
  統合ユーザ管理でのユーザ認証を実現するフレームワークです。Java 標準仕様（JAAS）に従ったアプリケーションプログラムインタフェースを提供します。

5.2.1　統合ユーザ管理の目的

従来，企業内にある各種の業務システムでは，業務の要件によってそれぞれ独自の方法でそのシステムのユーザを管理しています。例えば，勤休管理のシステムでは，社員データベースに登録された社員個人をシステムのユーザとして区別します。また，資材の発注システムでは，原価管理のための各部署をシステムのユーザとして区別します。最近はイントラネット技術の発展によって，システムごとのクライアントプログラムをインストールしなくても，Web ブラウザを使ってこれらの業務サービスを受けられるようになってきています。しかし，社内メールサービスを利用するにはメールの ID で，勤休管理サービスでは社員番号で，また資材発注サービスは部署コードでログインをするといったように，それぞれのシステムのユーザ管理に従ったユーザ認証のための操作が必要で，簡単にそれらのシステムのサービスを統合した新たなサービスを提供することは困難でした。

統合ユーザ管理フレームワークは，JavaEE の技術でこれらのサービスを統合するためのユーザ管理のフレームワークです。統合ユーザ管理フレームワークの例を次の図に示します。

図 5-1　統合ユーザ管理フレームワークの例



5.2.2　レルムを使用したユーザ管理とユーザマッピング

ここでは，統合ユーザ管理で使用する概念である，レルムとユーザマッピングについて説明します。

(1) レルム（Realm）

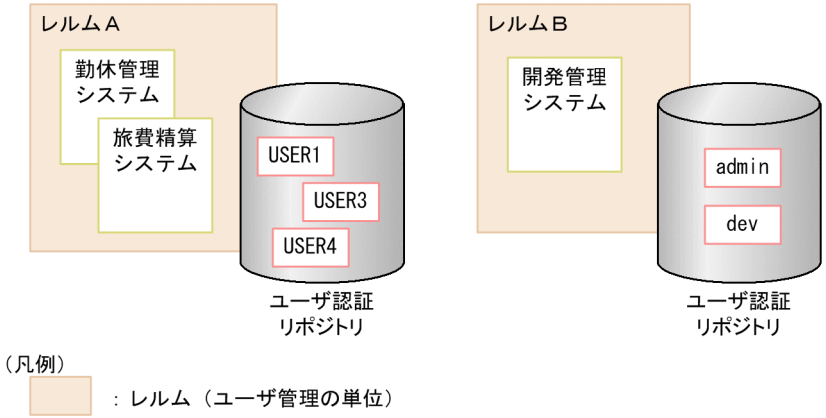
レルムとは，同一の認証ポリシーを適用する範囲のことです。業務サービスを提供するアプリケーションは，提供するサービスの要件に従って，そのサービスを利用するユーザを区別します。ユーザを区別するための処理は，一般に**認証処理**と呼ばれます。認証処理は，サービスを利用するユーザを区別するための認証メカニズムと，そのためのユーザ認証データベース（**ユーザ認証リポジトリ**）によって分類できます。どのような認証メカニズム（群）を利用し，どのようなユーザをユーザ認証データベースに登録するかといった認証ポリシーを決定するのは，システムを管理するシステム管理者の仕事です。

単一のサービスを提供するシステムを構築する場合にはあまり問題になりませんが，多数のサービスを提供するシステムを構築する場合，どの範囲でどのような認証ポリシーを適用してシステムを運用するかという検討が必要です。Web システムでは，同一の認証ポリシーを適用する範囲を**レルム**と呼び，それぞれのレルムを識別する名称のことを**レルム名**と呼びます。あるレルムで認証されたユーザは，該当レルム内でそのユーザを一意に識別する識別子（ユーザ ID）を持っています。

同じユーザ管理要件を持つ複数のアプリケーションは，一つのレルムで運用できます。新たにすべてのサービスを構築する場合，すべてのサービスを一つのレルムにして，認証されたユーザの持つセキュリティ属性によって各種のサービスに対してコントロールするようにできれば管理しやすいのですが，現実にはなかなかそのようなシステムの構築はされません。社内メールではメールの ID，人事システムでは従業員番号，資材システムには部署コードでログインするといったように，企業システムにはすでに多くのレルムがあり，これらは別々に管理運用されています。

管理者は，Application Server を使用してこれらのサービスの統合をする場合，何のためにそのレルムが必要なのか分析し，運用管理を簡単にするためにできるだけ少ないレルムで運用できるように，レルムの統廃合を検討してください。レルムの運用例を次の図に示します。

図 5-2　レルムの運用例



(2) レルム間のユーザマッピング

業務サービスを提供するアプリケーションは，ユーザの情報が必要になると，まずエンドユーザにユーザ ID とパスワードの入力を求め，それを基にユーザを認証します。一度受けた認証の結果は，ログアウトするまで保持されます。ただし，ユーザ ID およびパスワードが異なるアプリケーションに続けてログインしようとすると，そのたびにアプリケーションからユーザ ID とパスワードの入力を求められます。つま

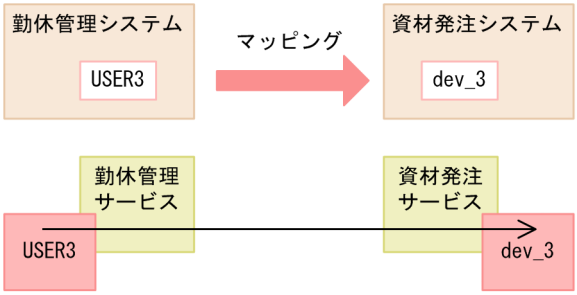
り，一度一つのアプリケーションで認証を受けたユーザであっても，異なるレルムで管理されているアプリケーションにアクセスするためには，そのつど認証を受ける必要があります。

これに対して，あるレルムでログインしたユーザが，ほかのレルムでどのユーザとして扱われるのかが分かれば，レルムが異なる J2EE アプリケーションにログインするたびにユーザ ID とパスワードを入力する必要はなくなります。統合ユーザ管理フレームワークでは，この問題を解決するために，あるレルムでログインしたユーザをほかのレルムのユーザにマッピングする**ユーザマッピング**を実現します。

ユーザマッピングとは，元のレルムでの認証が成功したという認証状態が引き継がれていれば，自動的にほかのレルムにマッピングしてユーザ認証をするものです。ユーザマッピングをする場合，Application Server のシステム管理者が，あるレルムのユーザとほかのレルムのユーザとのマッピング情報をあらかじめシステムに定義しておきます。

次の図に示すようなユーザマッピングの例では，勤休管理サービスで認証されたユーザ「USER3」に対して，資材発注サービスのユーザ「dev_3」のマッピングを定義しておきます。これによって，勤休管理サービスに「USER3」として認証されたユーザは，資材発注サービスへのログイン操作をしなくても自動的に該当サービスに対して「dev_3」として認証されます。

図 5-3　ユーザマッピングの例



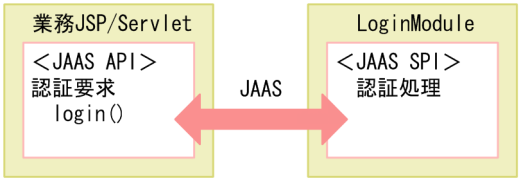
5.2.3　Java 標準仕様（JAAS）に準拠したユーザ認証の概要

今までの業務システムの Web 化では，従来の業務システムが持っていたユーザ認証機構を Web のサービスから独自のインタフェースで呼び出していました。また，最初から JavaEE の技術技術で構築される Web 業務サービスも，各社のアプリケーションサーバが提供する独自のユーザ認証機構を使って構築されてきました。これらのインタフェースの違いは，サービスを統合する場合の大きな障害になっていました。しかし，Java の標準ユーザ認証の仕様として JAAS（Java Authentication and Authorization Service）1.0 が決まり，現在では JavaEE の技術で構築される Web 業務サービスのユーザ認証の標準となっています。

(1)　統合ユーザ管理フレームワークと個別のユーザ管理の関連

JAAS では，アプリケーションが認証を要求する側のインタフェース（API：アプリケーションプログラミングインタフェース）と，その要求を受け認証処理をする側のインタフェース（SPI：サービスプロバイダインタフェース）を規定しています。認証処理をする側のモジュールをログインモジュールといいます。API と SPI の関係を次の図に示します。

図 5-4　API と SPI の関係

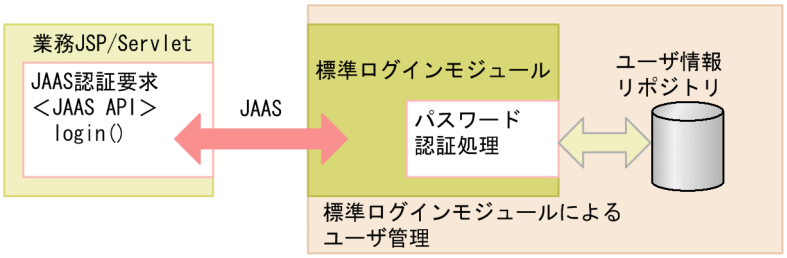


Application Server では，標準でパスワード認証の機能を持つ**標準ログインモジュール**を提供しています。この標準ログインモジュールを利用して，JAAS の API を使用する JSP/サーブレットで作成された J2EE アプリケーションのユーザ管理ができます。

標準ログインモジュールを使用すれば，アプリケーション開発者は，独自に認証モジュールを開発する必要はありません。また，モジュールはスタック可能なので，標準ログインモジュールと連携する拡張認証モジュールを容易に追加できます。さらに，完全に独自認証機能を必要とするアプリケーションでは，容易に標準ログインモジュールを独自認証モジュールに置き換えて使用できます。これによって，JAAS を使用してユーザ認証をしているアプリケーションプログラムを簡単に Application Server で統合できるようになります。

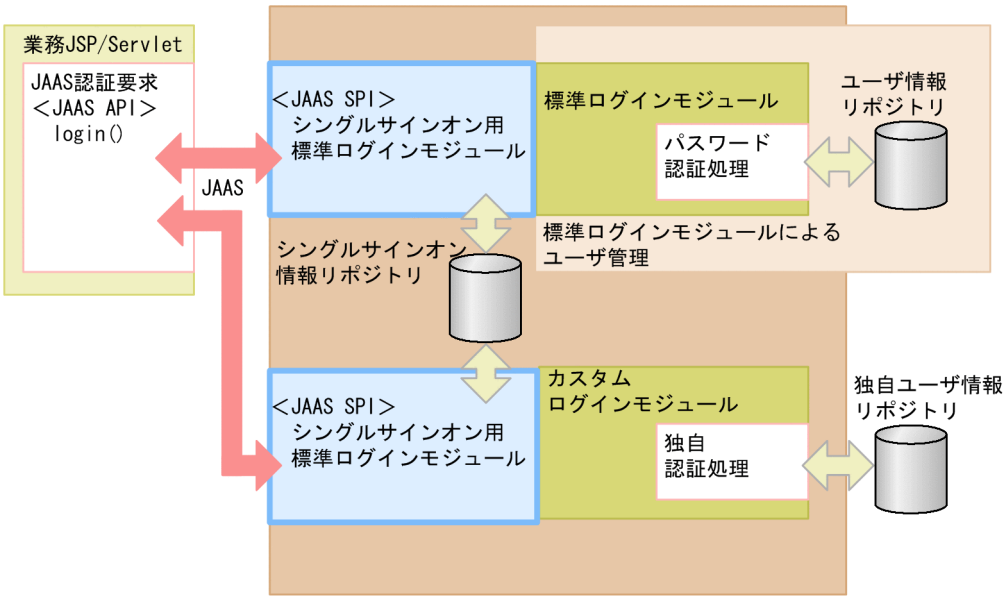
これを，JAAS 対応ユーザ管理と呼びます。JAAS 対応ユーザ管理の概要を次の図に示します。なお，図中の**ユーザ情報リポジトリ**とは，認証処理に必要なユーザ情報が格納されているリポジトリです。

図 5-5　JAAS 対応ユーザ管理の概要



統合ユーザ管理フレームワークに従って，JAAS 対応ユーザ管理を使用すると，ある業務サービスで認証されたユーザを，別の業務サービスのユーザにマッピングして認証要求する，ユーザマッピングの機能を利用できます。また，独自のユーザ情報リポジトリを利用する場合でも，カスタムログインモジュールを作成すれば，その業務サービスも含めたシングルサインオンが実現できます。ユーザマッピングによるシングルサインオンの概要を次の図に示します。なお，図中の**シングルサインオン情報リポジトリ**とは，シングルサインオンに必要なマッピング情報が格納されているリポジトリです。

図 5-6　ユーザマッピング機能の概要



(2)　標準ログインモジュールの概要

標準ログインモジュールは，パッケージ javax.security.auth.spi に含まれる LoginModule インタフェースの実装クラスです。実現する認証方法に応じて使い分けられます。

標準ログインモジュールの特徴を次に示します。

既存のユーザ情報（LDAP 情報またはデータベース情報）を利用したユーザ認証ができます。

標準ログインモジュールでは，ユーザ認証に使用するユーザ情報を格納するリポジトリとして LDAP ディレクトリサーバまたはデータベース（RDB）を使用できます。

LDAP ディレクトリサーバを使用する場合，標準のユーザ管理リポジトリの DIT（Directory Information Tree）の構造を Application Server が規定しています。また，すでに LDAP を導入しているときには，その情報を利用できるように簡単にカスタマイズできます。なお，DIT とは，ユーザや組織の情報をツリー構造で管理するための LDAP の仕組みです。詳細については，「5.2.4　統合ユーザ管理で使用するユーザ情報の管理方法」を参照してください。

認証方式としては，証明書を使用した認証やパスワード認証ができます。認証方式は，標準ログインモジュールの種類によって異なります。

ユーザ情報を参照できます。

ユーザ認証が成功した場合に，ログインユーザについての情報を参照できます。

JAAS では，認証が成功すると認証したユーザの情報を Credential として Subject に設定することが規定されています。また，これを要求元のアプリケーションから java.util.set インタフェースで参照するための一般的なメソッド（getPublicCredentials メソッドおよび getPrivateCredentials メソッド）が規定されています。

標準ログインモジュールでは，ユーザ情報を参照するためのインタフェースを提供しています。このインタフェースでは，コンフィグレーションに従って，ユーザの情報を Application Server が提供する UserAttributes インタフェースで扱うオブジェクトとして，Credential に設定します。アプリケーショ

ンでは標準のインタフェースでこのオブジェクトを取り出し，該当オブジェクトの getAttribute メソッドに属性名を指定して呼び出すことで，必要なユーザ情報を取得できます。また，属性名の別名（エイリアス）を指定して，情報を取得することもできます。

シングルサインオンを実現できます。

標準ログインモジュールは，シングルサインオンに対応できます。

なお，シングルサインオンを実現する場合，ユーザ情報を管理しているリポジトリの種類と関係なく，LDAP ディレクトリサーバが必要です。

5.2.4　統合ユーザ管理で使用するユーザ情報の管理方法

ここでは，統合ユーザ管理で使用するユーザ情報の管理方法について説明します。

統合ユーザ管理では，ユーザ情報を格納するリポジトリとして，LDAP，またはデータベースを使用します。LDAP ディレクトリサーバでは，ユーザや組織の情報は，DIT で管理されます。統合ユーザ管理フレームワークで使用する LDAP ディレクトリサーバでは，ユーザおよびレルムを DIT のエントリとして管理します。エントリは，DIT を構成する情報で，DIT の各節に該当する情報です。個々のエントリは，DN（識別名）で区別されます。

Application Server では，統合ユーザ管理フレームワークで使用する LDAP ディレクトリサーバに格納する標準のユーザ管理リポジトリの DIT 構造を規定しています。統合ユーザ管理フレームワークで使用するリポジトリは，2 種類あります。

- **ユーザ情報リポジトリ**

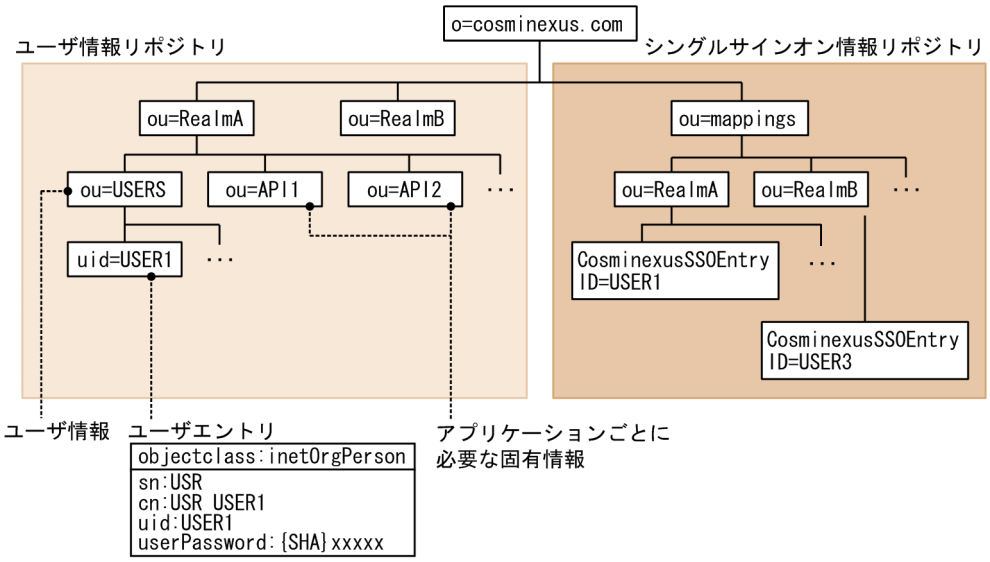
  ユーザ情報を格納します。

- **シングルサインオン情報リポジトリ**

  統合ユーザ管理フレームワークのユーザマッピングを実行して，シングルサインオンでユーザ認証するための各システムの認証情報とマッピング情報を格納します。

これらのリポジトリは，次の図に示すようなディレクトリ構造を持ちます。

図 5-7　統合ユーザ管理フレームワークのリポジトリの DIT 構造

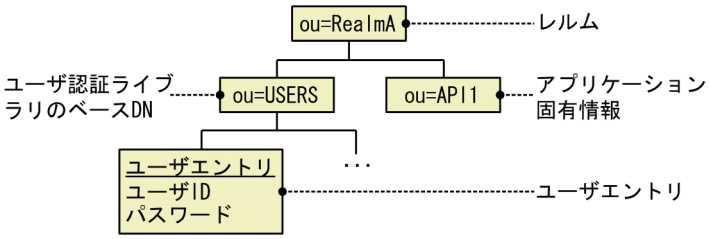


各リポジトリについて説明します。

(1)　ユーザ情報リポジトリ（LDAP ディレクトリサーバの場合）

ユーザ認証で使用するユーザ情報は，ユーザ情報リポジトリに格納します。統合ユーザ管理フレームワークは，LDAP ディレクトリサーバのユーザ情報リポジトリに格納されたユーザ情報を基にユーザを認証して，認証したユーザの情報をアプリケーションに渡します。ユーザ情報リポジトリのユーザ情報の参照には，ユーザ認証ライブラリを使用します。ユーザ情報リポジトリの DIT 構造を次の図に示します。

図 5-8　ユーザ情報リポジトリの DIT 構造



管理するレルムごとにユーザ情報リポジトリを設定してください。

(a)　レルム

JAAS 対応ユーザ管理のレルム名を指定します。レルム名は次の表に示す基準に従って指定してください。

表 5-2　レルム名の基準

情報の種類	意味	文法
レルム名	ユーザ管理の範囲を示す識別子。	英数字列。 大文字と小文字を区別しません。 DN 名で使用できる名前を付けてください。

注　英数字列は，英字（A～Z，a～z）と数字（0～9）の文字の並びを意味します。ASCII 文字列で指定してください（文法については，プログラムではチェックしていません）。

(b)　アプリケーション固有情報

該当レルムを使用する各アプリケーション固有の情報を格納する場合に使用してください。統合ユーザ管理フレームワークを使用する場合に必須な情報はありません。

(c)　ユーザ認証ライブラリのベース DN

レルムに属するユーザエントリの上位エントリです。レルムに属する各ユーザのエントリは，このエントリの下階層にある必要があります。なお，ユーザエントリがこのエントリの直下にない場合は，ua.conf（統合ユーザ管理のコンフィグレーションファイル）の com.cosminexus.admin.auth.ldap.search.scope オプションを変更する必要があります。また，このエントリで指定した情報は jaas.conf（JAAS のコンフィグレーションファイル）に指定する必要があります。各コンフィグレーションファイルについては，「14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）」を参照してください。

(d)　ユーザエントリ

ユーザ情報を定義します。ユーザ認証ライブラリでは，ユーザ情報として次の表に示す属性が必要です。

表 5-3　ユーザ情報に必要な属性

属性名	説明	必須
ユーザ ID	ユーザ ID を格納します。属性は，文字列（cis など）である必要があります。デフォルトでは，「uid」属性名を使用します。	必須
パスワード	パスワードを格納します。属性はバイナリです。平文または暗号化した値を格納します。この属性に値が指定されていない場合は，アカウントが無効になります。デフォルトでは，「userPassword」属性名を使用します。	オプション
そのほかの属性	各アプリケーションが定義している属性です。	各アプリケーションの仕様に従ってください。

ユーザ ID およびパスワードの属性名は，jaas.conf（JAAS のコンフィグレーションファイル）で変更できます。

(e)　注意事項

ユーザ情報リポジトリのディレクトリ構成は，JAAS 対応ユーザ管理で推奨する DIT 構成です。別の構成で管理する場合は「ユーザ認証ライブラリのベース DN」以下の次の条件を満たすユーザのエントリが必要です。

- ユーザ ID とパスワードは，同一オブジェクトクラス内にある必要があります。
- パスワードの属性はバイナリである必要があります。また，暗号化した値が格納されていることを推奨します。平文で格納されている場合は，一度 ldif 形式のファイルに出力してから convpw コマンドで

暗号化して，再登録してください。convpw コマンドの詳細については，「convpw（パスワードの暗号化）」を参照してください。

- ユーザ ID およびパスワードは，同じ属性名を使用してマルチバリューで指定できますが，アプリケーションサーバシステムの運用上，一つのオブジェクトクラス内に一つだけ指定してください。もし，複数指定した場合は，最初に見つかったものを使用します。
- ユーザ ID として使用する属性のユーザ ID は，レルム内（ユーザ情報リポジトリのベース DN 下）で一意な値にしてください。

なお，ユーザ情報リポジトリのベース DN，使用するユーザ ID およびパスワードの属性名については，ua.conf（統合ユーザ管理のコンフィグレーションファイル）で指定します。ua.conf については，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。

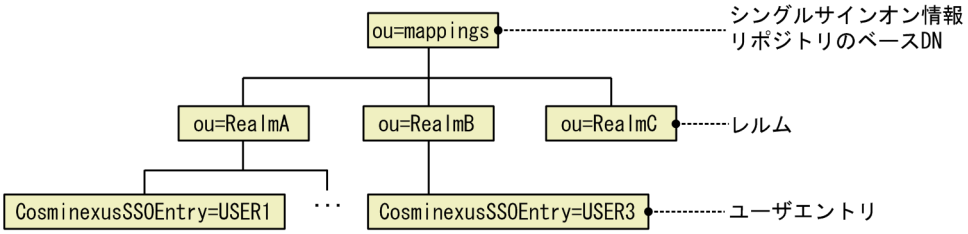
(2)　ユーザ情報リポジトリ（データベースの場合）

統合ユーザ管理フレームワークは，データベースに格納されたユーザ情報を基にユーザを認証できます。データベースでは，ユーザ ID からパスワードを取得できるようにしてください。

(3)　シングルサインオン情報リポジトリ

シングルサインオンでユーザ認証するための各システムの認証情報とマッピング情報は，シングルサインオン情報リポジトリに格納します。統合ユーザ管理フレームワークは，LDAP ディレクトリサーバのシングルサインオン情報リポジトリに格納されたユーザ情報を基にユーザマッピングをして，シングルサインオンを実現します。シングルサインオン情報リポジトリのユーザ情報は，シングルサインオンライブラリを使用して参照します。シングルサインオン情報リポジトリの DIT 構造を次の図に示します。

図 5-9　シングルサインオン情報リポジトリの DIT 構造



(a)　シングルサインオン情報リポジトリのベース DN

シングルサインオンを使用するために必要な情報を管理する DIT の，最上位のエントリです。このエントリは，ua.conf（統合ユーザ管理のコンフィグレーションファイル）で指定します。ua.conf については，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。なお，ここで指定する名称は，大文字と小文字が区別されません。この値は，標準のオブジェクトクラスである「organizationalUnit」の ou 属性に設定されます。

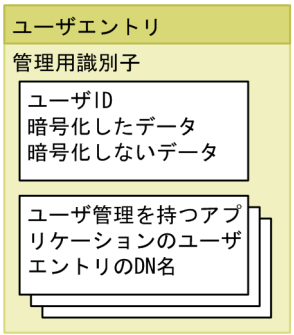
(b)　レルム

ユーザ情報はレルムごとに管理されます。シングルサインオン情報リポジトリのレルム名では，大文字と小文字は区別されません。この値は，標準のオブジェクトクラスである「organizationalUnit」の ou 属性に設定されます。

(c)　ユーザエントリ

シングルサインオンで接続したいアプリケーションで，ユーザ管理に使用しているユーザの認証情報および接続先を格納するためのエントリです。ユーザエントリの構造を次の図に示します。

図 5-10　ユーザエントリの構造



管理用識別子

シングルサインオンライブラリで登録した際に自動的に設定される識別子です。

ユーザ ID

各レルムで一意なユーザ ID を文字列で指定します。ユーザ ID では，大文字と小文字は区別されます。

暗号化したデータ

登録する際に暗号化が必要なデータを格納します。例えば，パスワードは，この属性に格納することで，暗号化されて保存されます。

暗号化しないデータ

登録する際に暗号化する必要がない，ユーザ ID および暗号化データ以外に認証に必要な情報を格納します。例えば，ユーザに付くグループ ID を保存します。

ユーザ管理を持つアプリケーションのユーザエントリの DN 名

ここには，ユーザが接続する，ユーザ管理を持つアプリケーションのユーザ認証情報の格納先（DN 名）が格納されます。この値は，複数指定できます。

5.2.5　ユーザ認証の有効期間と認証状態の引き継ぎ

JAAS でのユーザ認証の有効期限は，login メソッドの成功から logout メソッドを呼び出すまでです。

J2EE の Web アプリケーションでは，ユーザとの仮想的なセッションを HttpSession オブジェクトでコントロールします。複数の HTTP プロトコルの通信を一連のセッションとして扱う場合は，Web アプリケー

ションで Cookie または URL 書き換えなどで HttpSession オブジェクトと要求ユーザの結び付けをするようにコーディングします。

統合ユーザ管理フレームワークでは，ユーザ認証に成功した状態を HttpSession オブジェクトで記憶しています。同一 HttpSession オブジェクトを利用した要求で，かつ同一レルムの場合に限り，2 回目以降のログインでは，自動的に最初のログインで指定されたユーザの認証情報（ユーザ ID／パスワード）を使用して認証状態の引き継ぎをするため，エンドユーザからの認証情報の入力を省略できます。

ただし，次の表に示す順序でログインモジュールが使用された場合，エンドユーザから認証情報の入力を省略する機能は働きません。それぞれのログインモジュールの機能については，「5.3　標準ログインモジュールによるユーザ認証の仕組み」を参照してください。

表 5-4　エンドユーザから認証情報の入力を省略する機能が働かない使用順序

使用順序	使用するログインモジュール
1 回目のログイン	• DelegationLoginModule（カスタムログインモジュールや WebCertificateLoginModule を呼び出す場合） • WebCertificateLoginModule • WebSSOLoginModule（カスタムログインモジュールや WebCertificateLoginModule を呼び出す場合）
2 回目以降のログイン	• WebPasswordLDAPLoginModule

また，次の表に示す順序でログインモジュールを使用する場合，WebPasswordLDAPLoginModule による 2 回目以降のログインで認証情報の入力を省略するためには，最初のログインで使用するログインモジュールでパスワードを統合ユーザ管理のセッションに保持する必要があります。さらに，WebPasswordLDAPLoginModule での 2 回目以降のログインに使用するパスワードが最初のログインで保持したパスワードと同じでなければなりません。パスワードを保持するかどうか，および保持する場合に暗号化をするかどうかは，jaas.conf（JAAS のコンフィグレーションファイル）および ua.conf（統合ユーザ管理のコンフィグレーションファイル）で指定できます。jaas.conf（JAAS のコンフィグレーションファイル）の指定方法については，「14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）」を参照してください。ua.conf（統合ユーザ管理のコンフィグレーションファイル）の指定方法については，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。

表 5-5　2 回目以降のログインで認証情報の入力を省略するために条件がある使用順序

使用順序	使用するログインモジュール
1 回目のログイン	• WebPasswordLoginModule • WebPasswordJDBCLoginModule • WebPasswordLDAPLoginModule
2 回目以降のログイン	• WebPasswordLDAPLoginModule

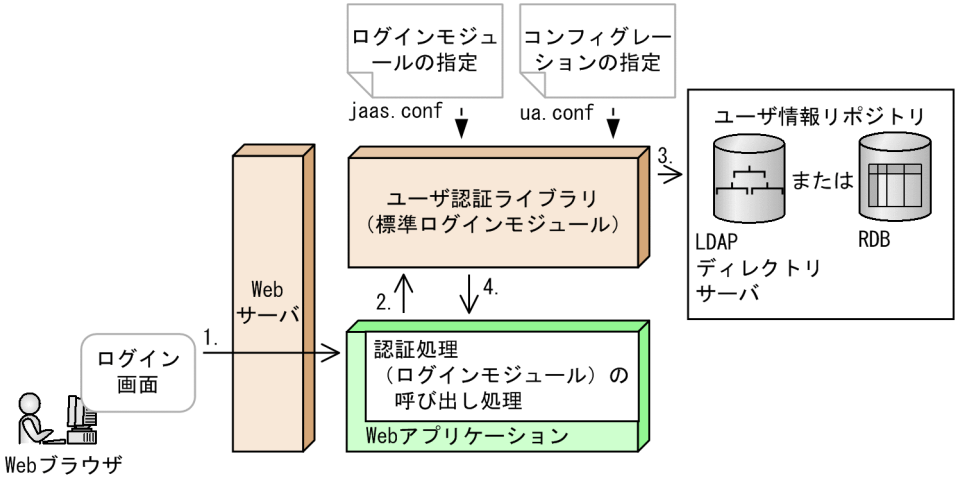
保持するパスワードを暗号化しない場合，J2EE サーバが使用するメモリの内容を何らかの方法で参照されるとパスワードが漏洩する可能性があります。また，セッションフェイルオーバ機能を使用する場合，パスワードを含むセッション情報がネットワークを流れるため，パスワードが盗聴されるおそれがあります。

暗号化をすることで，パスワードが漏洩するおそれは少なくなりますが，性能は劣化します。暗号化をするかどうかは，必要とするセキュリティや性能に応じて決定してください。なお，暗号化する場合，暗号アルゴリズムにはトリプル DES が使用されます。

5.2.6　統合ユーザ管理の処理の流れ

統合ユーザ管理を使用する場合の処理の流れを次の図に示します。

図 5-11　統合ユーザ管理を使用する場合の処理の流れ



図の流れについて説明します。

1. Web ブラウザからログインを要求するリクエストを送信します。
2. Web アプリケーションから，認証処理をするためのログインモジュールの呼び出し処理が実行されます。
3. 標準ログインモジュールを使用して，ユーザ認証処理が実行されます。使用するログインモジュールおよびそのコンフィグレーションは，jaas.conf（JAAS のコンフィグレーションファイル）または ua.conf（統合ユーザ管理のコンフィグレーションファイル）に定義されています。また，認証に必要な情報は，ユーザ情報リポジトリである LDAP ディレクトリサーバまたはデータベースから取得します。
4. 認証結果が Web アプリケーションに返却されます。

この処理を実現するためには，システムを構築するユーザとアプリケーション開発者によるシステムの設定およびアプリケーションの開発が必要です。

システムを構築するユーザの作業の概要

- jaas.conf（JAAS のコンフィグレーションファイル）および ua.conf（統合ユーザ管理のコンフィグレーションファイル）に，使用するログインモジュール，使用するリポジトリ，およびそれらのコンフィグレーション情報を定義します。シングルサインオンを使用する場合には，ua.conf（統合ユーザ管理のコンフィグレーションファイル）にシングルサインオン用のパラメタも定義します。
- シングルサインオンを使用する場合には，暗号鍵ファイルを作成します。
- ユーザ情報リポジトリにユーザ情報を登録しておきます。

- UNIX の場合，Component Container 管理者および統合ユーザ管理グループに設定したユーザが，<Application Server のインストールディレクトリ>/manager/config 以下に格納されている jaas.conf（JAAS のコンフィグレーションファイル）および ua.conf（統合ユーザ管理のコンフィグレーションファイル）を使用するときは，それぞれのコンフィグレーションファイルにあらかじめ適切なアクセス権を設定しておいてください。

なお，jaas.conf および ua.conf はテキストエディタなどで編集してください。

統合ユーザ管理機能の設定については，「5.19　コンフィグレーションファイルの作成」を参照してください。

アプリケーション開発者の作業の概要

- JAAS の API および Application Server が提供している統合ユーザ管理 API，JSP タグライブラリを使用して，ログインモジュールを呼び出す認証処理プログラムを作成します。
- それぞれのアプリケーションで独自の認証処理をする場合は，カスタムログインモジュールを作成します。
- パスワード認証時のパスワード拡張をする場合は，実装クラスを作成します。

統合ユーザ管理フレームワークを使用したユーザ認証の実装については，「5.9　統合ユーザ管理フレームワークによるユーザ認証の実装」を参照してください。

5.3　標準ログインモジュールによるユーザ認証の仕組み

統合ユーザ管理フレームワークでは，JAAS に準拠した標準ログインモジュールを提供します。標準ログインモジュールを使用することで，独自に認証モジュールを作成しなくても，アプリケーションサーバシステムのユーザの統合管理が実現できます。

5.3.1　標準ログインモジュールの種類と機能

統合ユーザ管理フレームワークが提供する標準ログインモジュールは，次の 2 種類に分類できます。

- ユーザを認証するために使用するログインモジュール

  次の四つのログインモジュールが該当します。

  - WebPasswordLoginModule

    パスワードを使用してユーザ認証するログインモジュールです。

  - WebCertificateLoginModule

    クライアント証明書を使用してユーザ認証するログインモジュールです。

  - WebPasswordLDAPLoginModule

    LDAP ディレクトリサーバの認証機能を使用してユーザ認証するログインモジュールです。

  - WebPasswordJDBCLoginModule

    ユーザ情報リポジトリとして，すでにデータベースを使用している場合に使用するログインモジュールです。

- 各アプリケーションのカスタムログインモジュールを呼び出すためのログインモジュール

  次の二つのログインモジュールが該当します。

  - DelegationLoginModule

    カスタムログインモジュールを呼び出す場合に使用するログインモジュールです。シングルサインオンには対応しません。

  - WebSSOLoginModule

    シングルサインオンで使用するログインモジュールです。標準ログインモジュールやカスタムログインモジュールなど，ほかのログインモジュールを呼び出します。

  DelegationLoginModule は，シングルサインオンの機能を使用しない場合にカスタムログインモジュールを呼び出すときに使用します。WebSSOLoginModule は，シングルサインオンの機能を利用する場合に，ほかの標準ログインモジュールまたはカスタムログインモジュールを呼び出すときに使用します。例えば，パスワードを使用してユーザ認証をしたいアプリケーションを，シングルサインオンの対象にしたい場合は，WebSSOLoginModule から WebPasswordLoginModule を呼び出して使用します。

それぞれのログインモジュールの機能一覧を次の表に示します。

表 5-6　ログインモジュールの機能一覧

機能				種類					
				P	C	L	J	D	S
使用するリポジトリ	LDAP ディレクトリサーバ			○	○	○	－	－	○
	データベース（JDBC でのアクセス）			－	－	－	○	－	－
認証方式	X509 証明書			－	○	－	－	－	－
	パスワード認証			○	－	○※1	○	－	－
		格納できる型	バイナリ（byte[]）	○	－	－	○※2	－	－
			文字列（String）	－	－	－	○※3	－	－
		比較／格納するときに利用できる暗号アルゴリズム	平文	○	－	－	○	－	－
			SHA-1 形式	○	－	－	○	－	－
			SHA-224 形式	○	－	－	○	－	－
			SHA-256 形式	○	－	－	○	－	－
			SHA-384 形式	○	－	－	○	－	－
			SHA-512 形式	○	－	－	○	－	－
			MD5 形式	○	－	－	○	－	－
			暗号化形式の拡張	○	－	－	○	－	－
			トリプル DES 形式	－	－	－	－	－	○
そのほか	Principal オブジェクトの設定			○	○	○	○	－	－
	ユーザ属性の取得			○	○	○	－	－	－
	ログイン時，統合ユーザ管理のセッションにログインしたユーザのユーザ ID／レルム名の登録（ログアウト時に削除）			○	○	○	○	○※4	○※4
	カスタムログインモジュールの呼び出し			－	－	－	－	○	○

（凡例）

P：WebPasswordLoginModule
C：WebCertificateLoginModule
L：WebPasswordLDAPLoginModule
J：WebPasswordJDBCLoginModule
D：DelegationLoginModule
S：WebSSOLoginModule
○：その機能を持ちます。
－：その機能を持ちません。

注※1　パスワードを格納できる型および暗号アルゴリズムの種類は，使用する LDAP ディレクトリサーバに依存します。
注※2　byte[]型にマッピングできる SQL のデータ型を指定できます（VARBINARY/LONGVARBINARY）。

注※3　String[]型にマッピングできる SQL のデータ型を指定できます（CHAR/VARCHAR/LONGVARCHAR）。
注※4　条件を満たしている場合に登録されます。

5.3.2　WebPasswordLoginModule

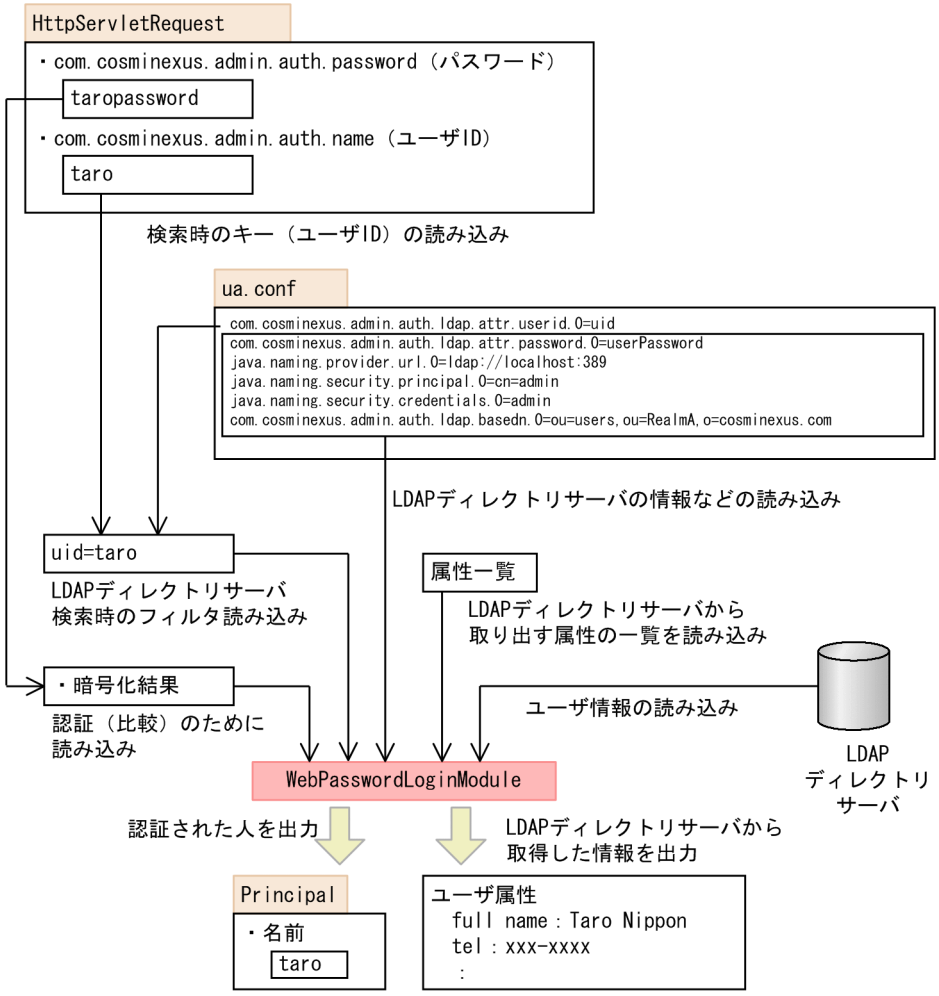
WebPasswordLoginModule は，LDAP ディレクトリサーバにユーザ情報リポジトリがある場合に，パスワード認証をするログインモジュールです。

入力されたユーザ ID/パスワードを使って，LDAP ディレクトリサーバに格納されているユーザ情報からパスワードを取得して認証します。

ua.conf（統合ユーザ管理のコンフィグレーションファイル）には，LDAP ディレクトリサーバに接続するための定義やエントリを検索する際に使用する属性名（uid および userPassword）を指定しておきます。

WebPasswordLoginModule ではその内容を読み取り，HttpServletRequest からユーザ ID を取得して LDAP ディレクトリサーバを検索し，パスワードを取得してパスワード認証をします。認証に問題がなければユーザ属性を返却します。WebPasswordLoginModule の概要を次の図に示します。

図 5-12　WebPasswordLoginModule の概要



5.3.3 WebCertificateLoginModule

WebCertificateLoginModule は，Web サーバで認証されたクライアント証明書を使用して認証をするログインモジュールです。

注意事項

統合ユーザ管理フレームワークの WebCertificateLoginModule を使用する場合は，SSL 機能を持つ Web サーバが必要になります。

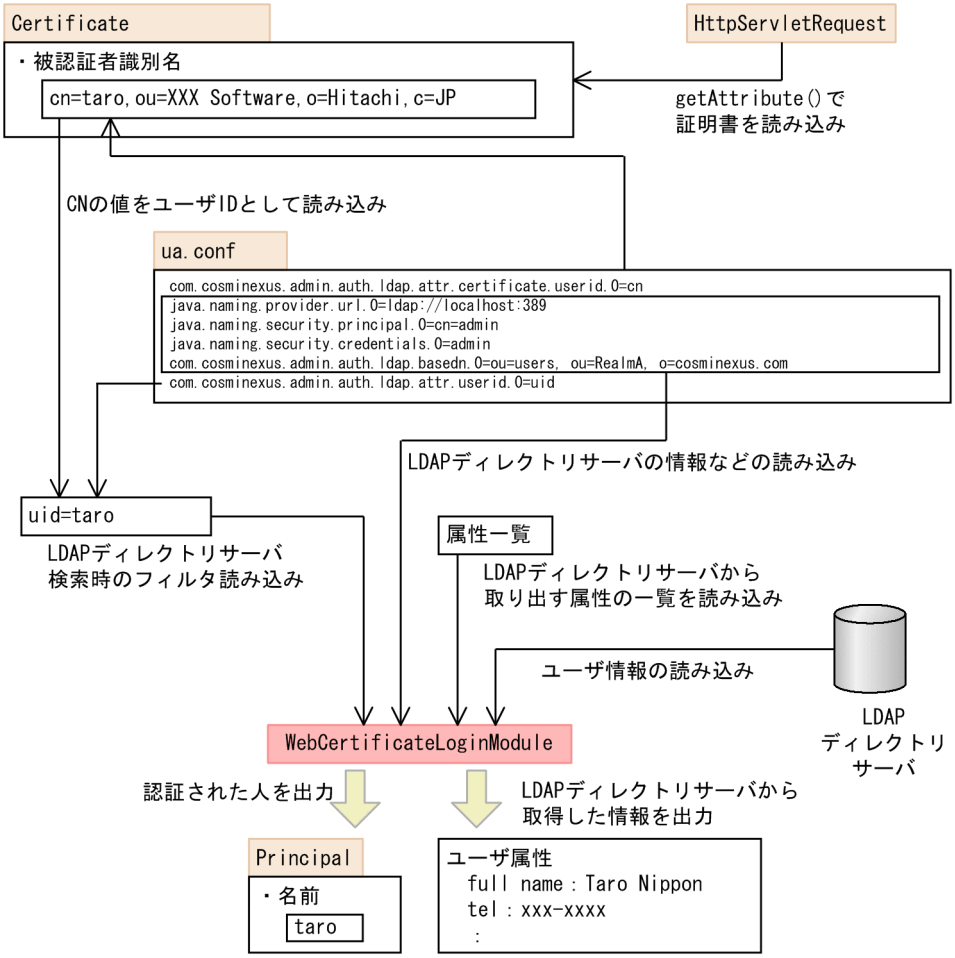
Web サーバが SSL 認証した際にブラウザに要求したクライアント証明書の被認証者識別名と，ユーザ情報リポジトリとでマッピングします。

ua.conf（統合ユーザ管理のコンフィグレーションファイル）には，クライアント証明書の被認証者の識別名内からユーザ ID に該当する属性名（cn）と LDAP ディレクトリサーバを検索する際に使用する属性名（uid）を指定しておきます。

WebCertificateLoginModule ではその内容を読み取り，クライアント証明書を使用して認証処理をします。クライアント証明書からユーザ ID に該当する ID を取得して LDAP ディレクトリサーバにアクセスした結果，認証に問題がなく，該当エントリがあればユーザ属性を返却します。なお，証明書内のユーザ ID に対応したエントリがない場合は，FailedLoginException 例外が発生します。

WebCertificateLoginModule の概要を次の図に示します。

図 5-13　WebCertificateLoginModule の概要

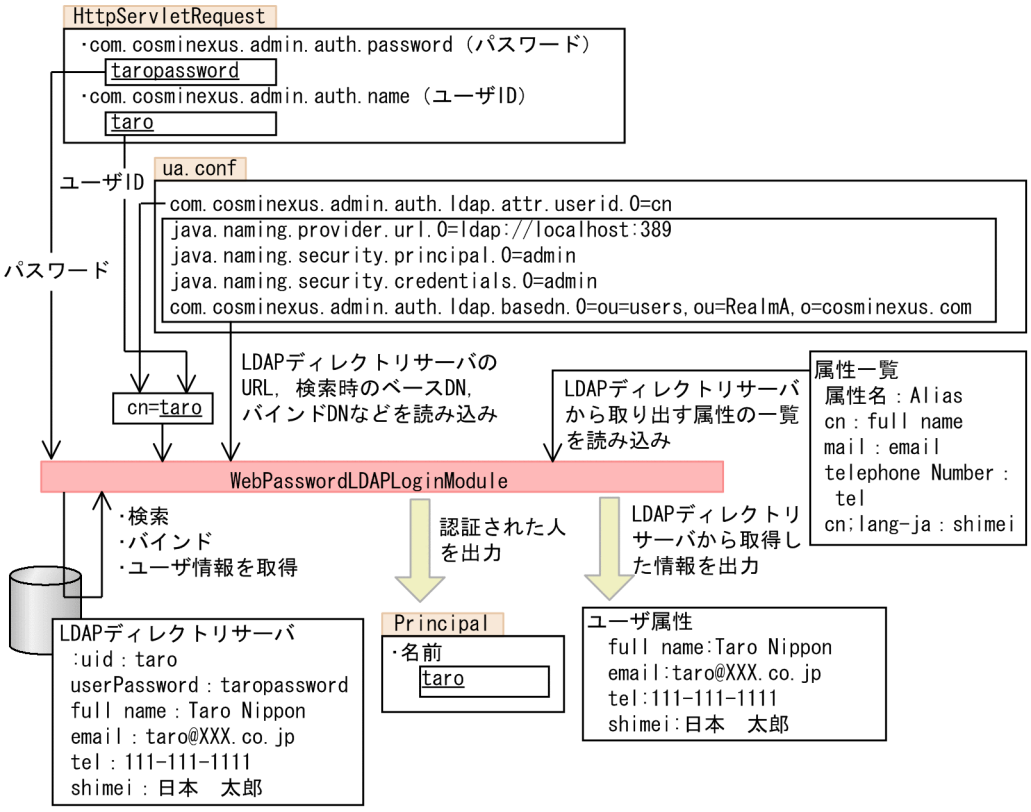


5.3.4　WebPasswordLDAPLoginModule

WebPasswordLDAPLoginModule は，LDAP ディレクトリサーバの認証機能を利用したログインモジュールです。

入力されたユーザ ID，パスワードを使って LDAP ディレクトリサーバにバインドし，バインドが成功すると認証が成功します。WebPasswordLDAPLoginModule の概要を次の図に示します。

図 5-14　WebPasswordLDAPLoginModule の概要



ua.conf（統合ユーザ管理のコンフィグレーションファイル）では，LDAP ディレクトリサーバに接続するための定義と，エントリを検索するときに使用する属性名を指定します。

WebPasswordLDAPLoginModule ではその内容を読み取り，HttpServletRequest からユーザ ID を取得して，ユーザ ID からユーザエントリの DN を求めます。次に，この DN と HttpServletRequest から取得したパスワードで LDAP ディレクトリサーバにバインドし，成功するとユーザ属性を返却します。

ユーザエントリ検索とユーザ属性の取得に使用するユーザ ID／パスワードについて

認証時にユーザエントリの検索をする場合，ua.conf（統合ユーザ管理のコンフィグレーションファイル）で設定されたバインド DN とバインド DN のパスワードを使用します。一方，ユーザ属性の取得に使用するバインド DN とパスワードには，ユーザエントリの DN とそのパスワードを使用します。ユーザエントリの検索については，「5.3.8(1)　ユーザエントリの検索」を参照してください。

LDAP 接続プールを利用する場合の注意

LDAP 接続プールはユーザエントリの検索処理だけに使用されます。認証時およびユーザ属性取得時に LDAP 接続プールは使用されません。したがって，検索をしない設定の場合は LDAP 接続プールを使用しない設定にしてください。LDAP 接続プールについては，「5.3.8(2)　接続プール」を参照してください。

5.3.5 WebPasswordJDBCLoginModule

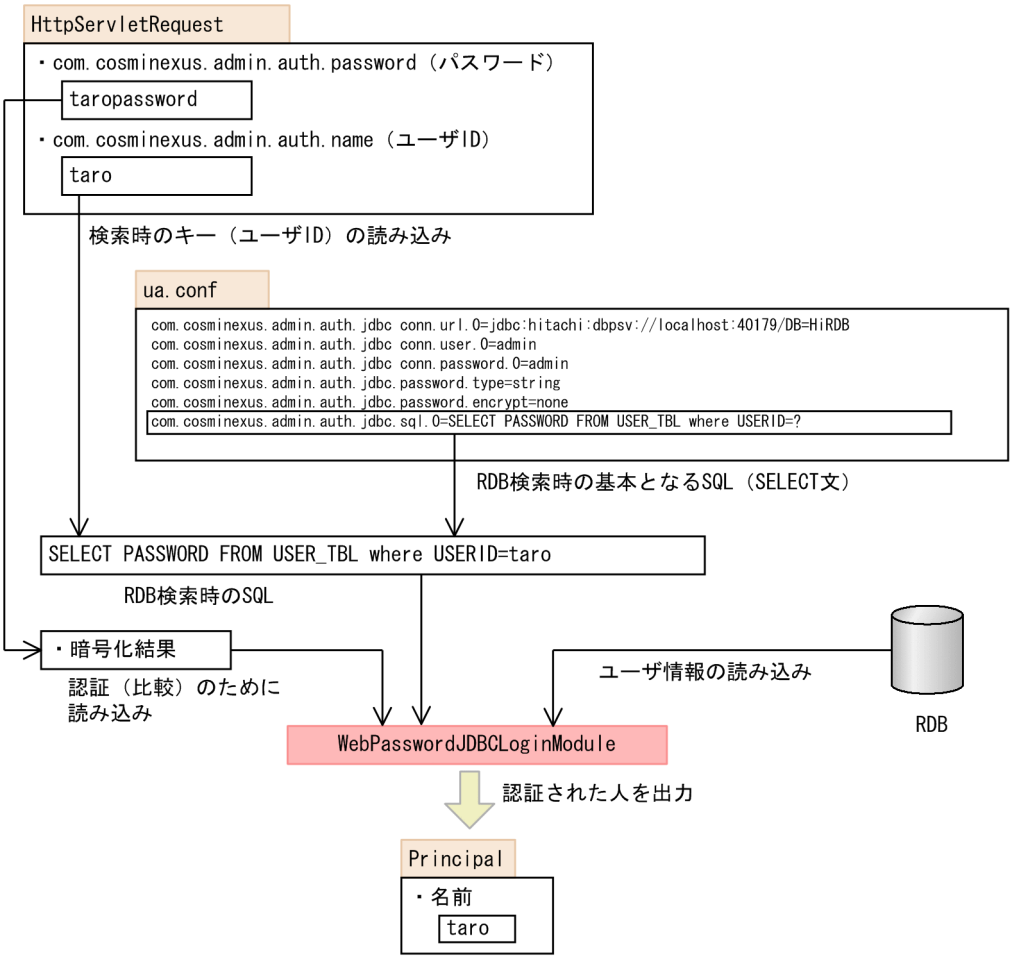
WebPasswordJDBCLoginModule は，すでにデータベースを用いてユーザ管理をしている場合に使用するログインモジュールです。

入力されたユーザ ID とパスワードを使って，データベース内に格納されているユーザ情報からパスワードを取得して認証します。

ua.conf（統合ユーザ管理のコンフィグレーションファイル）では，データベースに接続するための情報定義やエントリを検索する際に使用する SQL（SELECT 文）を指定します。

WebPasswordJDBCLoginModule ではその内容を読み取り，HttpServletRequest からユーザ ID を取得して JDBC を使ってデータベースにアクセスし，パスワードを取得してパスワード認証をします。WebPasswordJDBCLoginModule の概要を次の図に示します。

図 5-15 WebPasswordJDBCLoginModule の概要



また，WebPasswordJDBCLoginModule では，JDBC ドライバのクラスをログインモジュール内で参照します。使用できる JDBC ドライバと JDBC ドライバの設定手順を次に示します。

(1) 使用できる JDBC ドライバ

WebPasswordJDBCLoginModule で使用できるデータベースと JDBC ドライバを次の表に示します。

表 5-7 WebPasswordJDBCLoginModule で使用できるデータベースと JDBC ドライバ

データベース		JDBC ドライバ
HiRDB※		HiRDB Type4 JDBC Driver
Oracle	Oracle 11g	Oracle JDBC Thin Driver
SQL Server		SQL Server JDBC Driver

注※ HiRDB Run Time を含みます。

(2) JDBC ドライバの設定手順

JDBC ドライバのクラスを ua.conf（統合ユーザ管理のコンフィグレーションファイル）に設定します。また，JDBC ドライバを任意のディレクトリに格納し，その場所を J2EE サーバのクラスパスに追加してください。設定手順を次に示します。

1. ua.conf（統合ユーザ管理のコンフィグレーションファイル）に，次の内容を設定します。
   - 使用する JDBC ドライバに対応する JDBC ドライバのクラス名
   - データベースと接続するための URL
   - 代理で接続するデータベースユーザとそのユーザのパスワード

   データベースごとに設定例を次に示します。なお，太字部分はデータベースの環境に合わせて設定してください。

   HiRDB の場合

   com.cosminexus.admin.auth.jdbc.driver.0=JP.co.Hitachi.soft.HiRDB.JDBC.HiRDBDriver
   com.cosminexus.admin.auth.jdbc.conn.url.0=jdbc:hitachi:hirdb:// DBID=**22200**,DBHOST=**hostA**
   com.cosminexus.admin.auth.jdbc.conn.user.0=**system**
   com.cosminexus.admin.auth.jdbc.conn.password.0=**userpass**

   Oracle の場合

   com.cosminexus.admin.auth.jdbc.driver.0=oracle.jdbc.OracleDriver
   com.cosminexus.admin.auth.jdbc.conn.url.0=jdbc:oracle:thin:@**localhost:1521:orcl**
   com.cosminexus.admin.auth.jdbc.conn.user.0=**system**
   com.cosminexus.admin.auth.jdbc.conn.password.0=**userpass**

   SQL Server の場合

   com.cosminexus.admin.auth.jdbc.driver.0=com.microsoft.sqlserver.jdbc.SQLServerDriver

com.cosminexus.admin.auth.jdbc.conn.url.0=jdbc:sqlserver://
localhost:1433;DatabaseName=sqlserver

com.cosminexus.admin.auth.jdbc.conn.user.0=system

com.cosminexus.admin.auth.jdbc.conn.password.0=userpass

2. J2EE サーバが稼働するマシンの任意のディレクトリに JDBC ドライバの JAR ファイルを格納します。

3. J2EE サーバの usrconf.cfg（オプション定義ファイル）に，手順 2.で格納した JAR ファイルのパスを設定します。

   設定例を次に示します。

   add.class.path=<手順 2.で格納したディレクトリ>/<JAR ファイル名>

   なお，JAR ファイル名は接続するデータベースで異なります。

(3) 注意事項

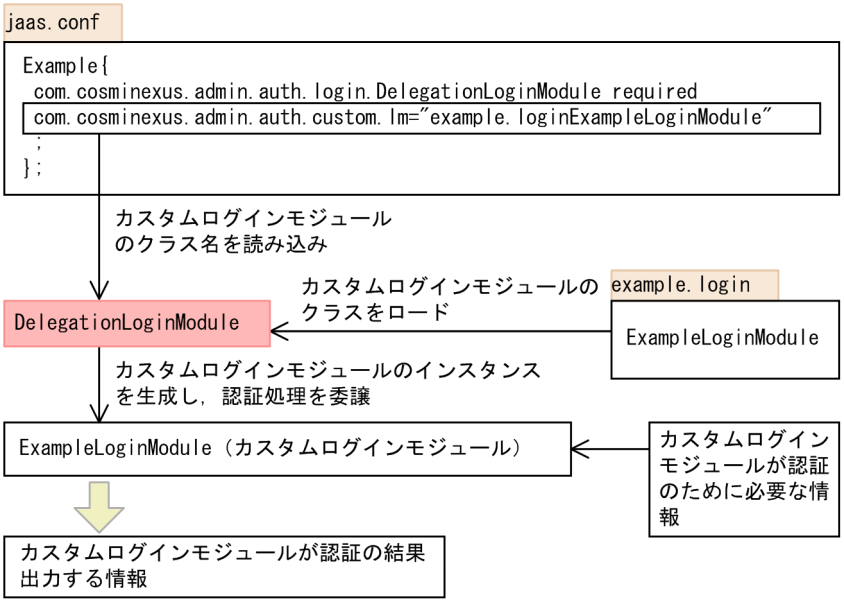
- Oracle の場合，データベース側で作成するユーザ情報用のテーブルのカラム（USERID など）に VARCHAR2 型などの可変長文字列を指定してください。CHAR 型などの固定長文字列にした場合は，パスワード認証に失敗することがあります。
- SQL Server への認証モードのうち，Windows 認証は使用できません。

5.3.6 DelegationLoginModule

カスタムログインモジュールを呼び出す場合に使用するログインモジュールです。

指定されたカスタムログインモジュールに認証処理を委譲します。DelegationLoginModule の概要を次の図に示します。

図 5-16 DelegationLoginModule の概要



jaas.conf（JAAS のコンフィグレーションファイル）では，使用するカスタムログインモジュールのクラス名を指定します。DelegationLoginModule はその内容を読み取り，カスタムログインモジュールをインスタンス化します。このとき，DelegationLoginModule の initialize メソッドに与えられた引数はそのままカスタムログインモジュールに渡されます。

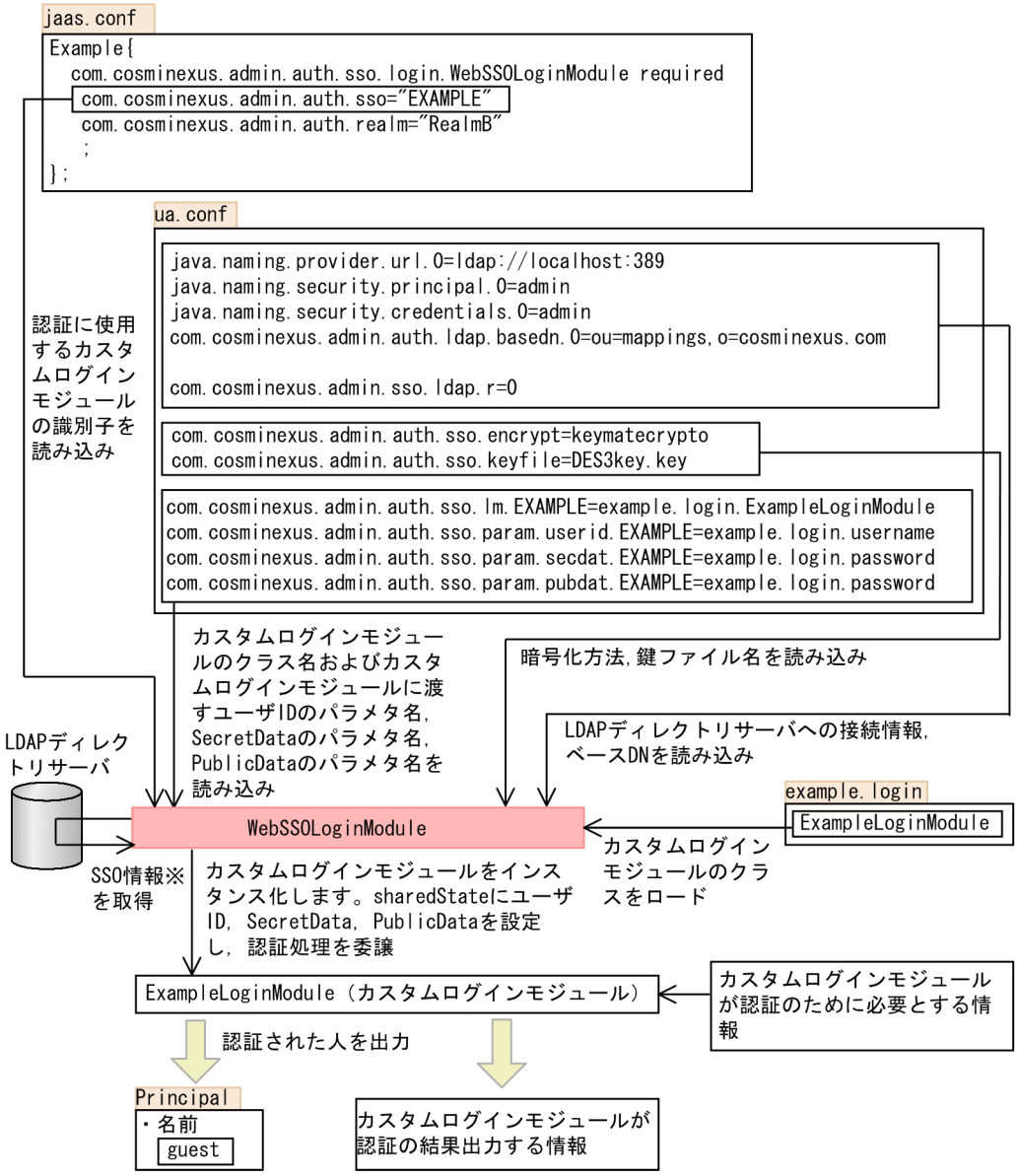
認証処理はカスタムログインモジュールに委譲されます。

5.3.7 WebSSOLoginModule

シングルサインオンをする場合に使用するログインモジュールです。標準ログインモジュールまたはカスタムログインモジュールを呼び出します。

一つのセッションですでにログインしているユーザが存在する場合，別のレルムで認証するための情報（ユーザ ID，SecretData，PublicData）をカスタムログインモジュールに渡します。
WebSSOLoginModule の概要を次の図に示します。

図 5-17　WebSSOLoginModule の概要



注※　シングルサインオン情報

WebSSOLoginModule は，jaas.conf（JAAS のコンフィグレーションファイル）で指定されたログインモジュールの識別子に対応するカスタムログインモジュールのクラス名を ua.conf（統合ユーザ管理のコンフィグレーションファイル）から読み込み，カスタムログインモジュールをインスタンス化します。このとき，WebSSOLoginModule の initialize メソッドに与えられた引数はそのままカスタムログインモジュールに渡されます。

該当するセッションでログインしているユーザが存在する場合，ua.conf で指定された LDAP ディレクトリサーバからログインしているユーザのシングルサインオン情報が取得されます。このシングルサインオン情報に，これからログインしようとしているレルムのユーザへのマッピング情報が存在する場合，マッピング先ユーザのシングルサインオン情報が取得されます。シングルサインオン情報のうち，SecretData は ua.conf で指定された方法で復号化されます。WebSSOLoginModule は，マッピング先ユーザのユーザ ID，復号化した SecretData，および PublicData を sharedState（カスタムログインモジュールに

initialize メソッドによって渡される Map オブジェクト）に設定します。設定に使用するパラメタ名は，ua.conf に指定します。

該当するセッションでログインしているユーザが存在しない場合，WebSSOLoginModule は sharedState を変更しません。

認証処理はカスタムログインモジュールに委譲されます。

5.3.8　標準ログインモジュールによるリポジトリへのアクセス

ここでは，標準ログインモジュールによるユーザ情報リポジトリへのアクセス方法について説明します。

(1)　ユーザエントリの検索

LDAP ディレクトリサーバをユーザ情報リポジトリとして使用する次のログインモジュールは，認証時にユーザエントリの検索ができます。

- WebPasswordLoginModule
- WebCertificateLoginModule
- WebPasswordLDAPLoginModule

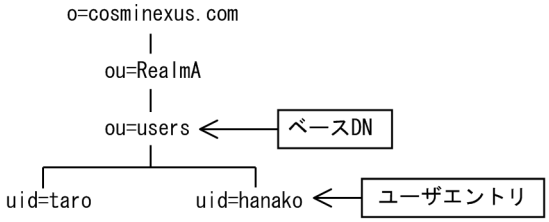
検索するかどうか，および検索範囲は，ua.conf（統合ユーザ管理のコンフィグレーションファイル）で設定します。なお，検索する必要があるかどうかは，LDAP ディレクトリサーバの DIT の構成によって決まります。

検索する必要がない場合

ユーザの認証およびユーザ属性を取得するためには，ユーザから入力されたユーザ ID を基に，LDAP ディレクトリサーバ上のユーザエントリを特定する必要があります。

次の図に示すように，ユーザエントリがベース DN の直下に存在し，かつユーザ ID がユーザエントリの RDN（相対識別名）に含まれていれば，ユーザエントリの DN は，ベース DN，ユーザ ID を表す属性名，およびユーザ ID から組み立てられるため，検索は不要です。統合ユーザ管理では，このように検索が不要となる DIT 構造にすることを推奨します。

図 5-18　ユーザエントリがベース DN の直下に存在する場合

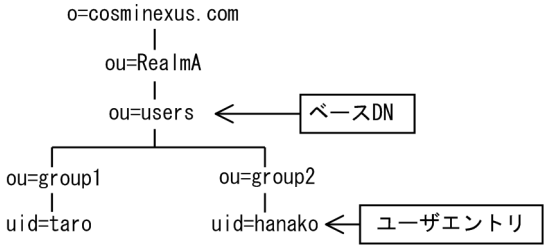


検索する必要がある場合

ユーザエントリの RDN にユーザ ID が含まれない場合，またはユーザエントリがベース DN の直下よりも下の階層に存在する場合は，検索しないとユーザエントリを特定できません。次の図に示すように

ユーザエントリがベース DN の 2 階層以上下層に存在する場合，検索範囲は subtree（ベース DN 以下すべての下層の検索）にする必要があります。

図 5-19　ユーザエントリがベース DN の 2 階層以上下層に存在する場合



(2)　接続プール

標準ログインモジュールでは，ユーザ情報リポジトリへのアクセスを高速化するために接続プールを使用できます。

接続プールについては，ua.conf（統合ユーザ管理のコンフィグレーションファイル）で設定します。

LDAP 接続プールを使用できるログインモジュールを次に示します。

- WebPasswordLoginModule
- WebCertificateLoginModule
- WebPasswordLDAPLoginModule
- WebSSOLoginModule

JDBC 接続プールを使用できるログインモジュールを次に示します。

- WebPasswordJDBCLoginModule

5.3.9　パスワード認証時の暗号拡張

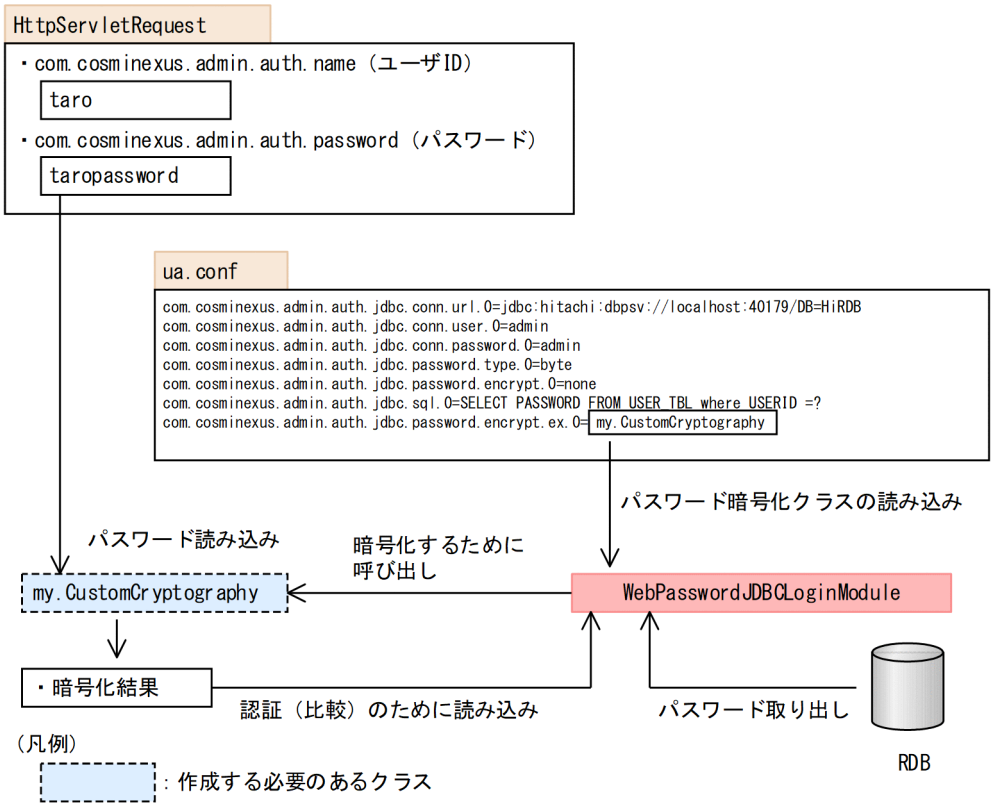
リポジトリに格納されているパスワードの暗号の形式が標準で用意されているもの（SHA-2/SHA-1/MD5/平文）以外の場合でも，暗号拡張ができます。この機能が利用できるのは，WebPasswordLoginModule と WebPasswordJDBCLoginModule です。

パスワード認証の暗号拡張をすると，リポジトリに格納されているパスワードの暗号の形式が標準で用意されている形式以外の場合でもパスワード認証をできるようにします。この機能を使うためには，あらかじめアプリケーション開発者が実装クラスを作成する必要があります。

ログインモジュールでは，データベースから取得したパスワードと比較するために HttpServletRequest から入力したパスワードを変換します。com.cosminexus.admin.auth.jdbc.password.encrypt.ex が ua.conf（統合ユーザ管理のコンフィグレーションファイル）に指定されていた場合，そのクラスの実装をインスタンス化して，HttpServletRequest から入力したパスワードを変換します。

変換したものとデータベースのパスワードを単純に byte[]で比較し，すべての文字が一致することで認証が成功します。パスワード認証時の暗号拡張の概要を次の図に示します。

図 5-20　パスワード認証時の暗号拡張の概要



実装クラスの作成方法については，「5.10　API を使用したユーザ認証の実装」を参照してください。

5.3.10　ログインモジュールが使用するコンフィグレーションファイルのパラメタ

使用する標準ログインモジュールの種類によって，ua.conf（統合ユーザ管理のコンフィグレーションファイル）に指定する必要があるパラメタが異なります。

(1)　LDAP ディレクトリサーバを使用するログインモジュール

LDAP ディレクトリサーバを使用する各ログインモジュールが使用するパラメタの一覧を次の表に示します。各パラメタの意味については，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。

表 5-8　LDAP ディレクトリサーバを使用する各ログインモジュールが使用するパラメタの一覧

パラメタ	種類			
	P	C	S	L
java.naming.provider.url	○	○	○	○
java.naming.security.principal	○	○	○	○※1
java.naming.security.credentials	○	○	○	○※1
com.cosminexus.admin.auth.ldap.basedn	○	○	○	○
com.cosminexus.admin.auth.ldap.attr.userid	○	○	×	○
com.cosminexus.admin.auth.ldap.search.userrdn	○	○	×	○※2
com.cosminexus.admin.auth.ldap.search.scope	○	○	×	○※3
com.cosminexus.admin.auth.ldap.attr.password	○	×	×	○※4
com.cosminexus.admin.auth.ldap.pool.enable	○	○	○	○※5
com.cosminexus.admin.auth.ldap.pool.max	○	○	○	○
com.cosminexus.admin.auth.ldap.pool.max_spare	○	○	○	○
com.cosminexus.admin.auth.ldap.pool.min_spare	○	○	○	○
com.cosminexus.admin.auth.ldap.pool.gc_interval	○	○	○	○
com.cosminexus.admin.auth.ldap.conn.retry.count	○	○	○	○
com.cosminexus.admin.auth.ldap.conn.retry.wait	○	○	○	○
com.cosminexus.admin.auth.ldap.certificate.attr.userid	×	○	×	×
com.cosminexus.admin.auth.ldap.password.encrypt	○	×	×	×
com.cosminexus.admin.auth.ldap.password.encrypt.ex	○	×	×	×
com.cosminexus.admin.auth.ldap.directory.kind	×	×	×	○

（凡例）

P：WebPasswordLoginModule

C：WebCertificateLoginModule

S：WebSSOLoginModule

L：WebPasswordLDAPLoginModule

○：使用できる　×：使用できない

注※1

ユーザエントリを検索する場合だけ必要です。このバインド DN とパスワードを使用して検索します。

注※2

ユーザエントリを検索する場合（ユーザエントリがベース DN の直下にない場合），true を設定してください。

注※3

ユーザエントリを検索する場合，subtree を指定してください。

注※4

ユーザのパスワードを変更する場合だけ必要です。使用する LDAP ディレクトリサーバが Active Directory の場合，unicodePwd を指定してください。Active Directory 以外の場合は userPassword を指定してください。

注※5

LDAP 接続プールはユーザエントリを検索する場合だけ使用されます。検索しない場合は false を指定してください。なお，ユーザエントリを検索する場合もしない場合も，ユーザ認証をするために LDAP ディレクトリサーバに接続するとき，LDAP 接続プールは使用されません。

(2)　データベースを使用するログインモジュール

データベースを使用する各ログインモジュールが使用するパラメタの一覧を次の表に示します。各パラメタの意味については，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。

表 5-9　データベースを使用する各ログインモジュールが使用するパラメタの一覧

パラメタ	種類
	J
com.cosminexus.admin.auth.jdbc.driver	○
com.cosminexus.admin.auth.jdbc.conn.url	○
com.cosminexus.admin.auth.jdbc.conn.user	○
com.cosminexus.admin.auth.jdbc.conn.password	○
com.cosminexus.admin.auth.jdbc.pool.enable	○
com.cosminexus.admin.auth.jdbc.pool.max	○
com.cosminexus.admin.auth.jdbc.pool.max_spare	○
com.cosminexus.admin.auth.jdbc.pool.min_spare	○
com.cosminexus.admin.auth.jdbc.pool.gc_interval	○
com.cosminexus.admin.auth.jdbc.conn.retry.count	○
com.cosminexus.admin.auth.jdbc.conn.retry.wait	○
com.cosminexus.admin.auth.jdbc.sql	○
com.cosminexus.admin.auth.jdbc.password.type	○

パラメタ	種類
	J
com.cosminexus.admin.auth.jdbc.password.encrypt	○
com.cosminexus.admin.auth.jdbc.password.encrypt.ex	○

（凡例）

J：WebPasswordJDBCLoginModule

○：使用できる

5.4　統合ユーザ管理のセッション管理

この節では，統合ユーザ管理で管理するセッションについて説明します。

5.4.1　セッションの種類

統合ユーザ管理の機能を使う上で存在するセッションとして考えられるものには次の種類があります。

- Web コンテナが管理するセッション（HttpSession）

  HttpSession オブジェクトのことを示します。

- 統合ユーザ管理のセッション

  標準ログインモジュールを使用してログインしてからログアウトするまでの期間を示します。

  一つのセッション内には複数のユーザがログインできるため，ログインしているすべてのユーザがログアウトしないとセッションは無効になりません。また，このセッションは Web コンテナが管理しているセッションである HttpSession と同期しています。そのため，HttpSession が無効になった場合は，ログインの有無に関係なくセッションは無効になります。

  統合ユーザ管理のセッションを制御する機能としては，次の 2 種類の機能があります。

  - 標準ログインモジュールでは，ログインしたユーザ ID とレルム名をセッションに自動で登録および削除できます。
  - 統合ユーザ管理はセッションフェイルオーバ機能に対応しています。統合ユーザ管理のセッションフェイルオーバ機能を有効にすることで，ログイン状態をセッションフェイルオーバ先に引き継げます。なお，セッションフェイルオーバ機能を使用しているときに，ログイン状態を引き継ぐかどうかは選択できます。

- カスタムログインモジュールのセッション

  各アプリケーションが個々に持つログインしているユーザのセッションを示します。このセッションの概念は，カスタムログインモジュールの仕様に依存します（通常はログインしてからログアウトまでの期間を示します）。

  なお，カスタムログインモジュールでは，ログインしたユーザ ID とレルム名をセッションに自動で登録および削除できます。

5.4.2　ログインしたユーザ ID の登録

ここでは，統合ユーザ管理のセッションにログインしたユーザ ID の登録について説明します。

(1)　ログインしているユーザ ID を登録する目的

統合ユーザ管理のセッションにログインしているユーザ ID を登録することで，次のようなことができます。

- JSP タグライブラリの<ua:notLogin>タグで，統合ユーザ管理のセッションでユーザがログインしているかどうかを判断できます。この指定は，任意のレルム名を指定することで，そのレルム内でのログインの有無も判断できます。

(2) ログインしているユーザ ID の登録条件

認証手段を持つ標準ログインモジュールでは，自動的に統合ユーザ管理のセッションにログインしたユーザ ID を登録します。

カスタムログインモジュールの場合，次の条件を満たしたログインモジュールでログインしたユーザ ID が，統合ユーザ管理のセッションに登録されます。

- カスタムログインモジュールの実装で，commit メソッドによって Principal オブジェクトが Subject に関連づけられていること
- 統合ユーザ管理フレームワークによって提供されている WebSSOHandler が，LoginContext クラスのコンストラクタの引数に指定されていること

もし，1 回の呼び出しで複数のログインモジュールを呼び出していた場合，認証手段を持つ標準ログインモジュールまたは上記の条件を満たしたログインモジュールでログインされるまでユーザ ID は登録されません。最後まで見つからなかった場合は，統合ユーザ管理のセッションにはログインしたユーザ ID は登録されません。

(3) 統合ユーザ管理のセッションに登録される内容

統合ユーザ管理のセッションに登録される内容は，レルム名，ユーザ ID およびログイン時刻の 3 種類です。

- **レルム名**
  jaas.conf（JAAS コンフィグレーションファイル）の com.cosminexus.admin.auth.realm で指定した値が設定されます。このオプションが省略されていた場合は空文字が仮定されます。なお，DelegationLoginModule 以外の場合，com.cosminexus.admin.auth.realm の指定は省略できません。
- **ユーザ ID**
  各ログインモジュールの commit メソッドを呼び出したあと，Subject から最初に求められた Principal オブジェクトのユーザ ID（getName メソッドの結果）が設定されます。
- **ログイン時刻**
  統合ユーザ管理のセッションにユーザ ID を登録時，最初にログインした時刻がユーザ単位に設定されます。

5.4.3 統合ユーザ管理のセッションに登録したユーザ ID の削除

認証した Subject を持つ LoginContext クラスの logout メソッドが呼ばれたとき，自動的に統合ユーザ管理のセッションに登録されたユーザ ID とレルム名が削除されます。

5.4.4　JAAS のコンフィグレーションファイルの定義例

jaas.conf（JAAS のコンフィグレーションファイル）の定義例を示します。

(1)　Application Server の標準ログインモジュールを使った定義例

JAAS のコンフィグレーションファイルで次のように定義されていた場合は，最初に実行した WebPasswordLoginModule が認証した際のユーザ ID およびレルム名「RealmA」で統合ユーザ管理のセッションに登録します。

```
Example03 {
  com.cosminexus.admin.auth.login.WebPasswordLoginModule required
                                                  //セッションに参加します
    com.cosminexus.admin.auth.realm="RealmA"
    com.cosminexus.admin.auth.ldap.r="0"
    com.cosminexus.admin.auth.ldap.w="1"
    ;
  com.cosminexus.admin.auth.login.DelegationLoginModule required
    com.cosminexus.admin.auth.custom.lm="my.login.MyLoginModule"
    my.login.useracctterm="acctTerm"
    ;
};
```

(2)　カスタムログインモジュールだけを使用した場合の定義例

JAAS のコンフィグレーションファイルで次のように定義されていた場合，最初に実行した MyLoginModule1 の commit メソッドで設定した Principal オブジェクトのユーザ ID（getName メソッドの値）を使って統合ユーザ管理のセッションにユーザ ID を登録します（DelegationLoginModule がユーザ ID を登録します）。

なお，「com.cosminexus.admin.auth.realm」については指定がないため，レルム名として空文字（""）を仮定します。

```
Example99 {
  com.cosminexus.admin.auth.login.DelegationLoginModule required
                                                 //セッションに参加します
    com.cosminexus.admin.auth.custom.lm="my.login.MyLoginModule1"
    ;
  com.cosminexus.admin.auth.login.DelegationLoginModule required
    com.cosminexus.admin.auth.custom.lm="my.login.MyLoginModule2"
    ;
};
```

ただし，MyLoginModule1 が Principal オブジェクトを Subject に設定していない場合は，MyLoginModule2 が統合ユーザ管理のセッションにユーザ ID を登録する対象になります。もし，MyLoginModule2 も Principal オブジェクトを設定しない場合は，統合ユーザ管理のセッションにユーザ ID を登録しません。

jaas.conf の詳細については，「14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）」を参照してください。

5.5　シングルサインオンの利用方法

この節では，シングルサインオンの利用方法について説明します。**シングルサインオン**とは，一度のログイン処理で，ユーザ ID が異なる複数のシステムをシームレスに使用できるようにするための機能です。

5.5.1　シングルサインオンをするために必要な手順

シングルサインオンを利用するためには，シングルサインオンを使用するすべてのカスタムログインモジュールおよび標準ログインモジュール（ユーザ認証をするログインモジュール）を WebSSOLoginModule で呼び出す必要があります。

シングルサインオンへの切り替えは jaas.conf（JAAS のコンフィグレーションファイル）で定義します。

例えば，シングルサインオンを適用する前は次のように WebPasswordLoginModule を使ってユーザ認証をしていたとします。

```
AP1 {
  com.cosminexus.admin.auth.login.WebPasswordLoginModule Requisite
    com.cosminexus.admin.auth.ldap.r="3"
    com.cosminexus.admin.auth.ldap.w="2"
    com.cosminexus.admin.auth.realm=XXXcompany;
};
```

シングルサインオンを適用するには次の背景色付きの太字に示す修正をします。

```
AP1 {
  com.cosminexus.admin.auth.sso.login.WebSSOLoginModule Requisite
    com.cosminexus.admin.auth.ldap.r="3"
    com.cosminexus.admin.auth.ldap.w="2"
    com.cosminexus.admin.auth.realm=XXXcompany;
};
```

シングルサインオンをするすべてのログインモジュールの定義は，この例で示すような修正で変更したあとに J2EE サーバを起動することで利用できます。

5.5.2　既存アプリケーションで行っているユーザ管理の適用

利用しているアプリケーションがすでにユーザ管理を行っている場合に，統合ユーザ管理のシングルサインオンを適用するには，次の表に示す条件を満たしている必要があります。

表 5-10　統合ユーザ管理のシングルサインオンの適用に必要な条件

LoginModule の有無	条件 1	条件 2	適用	適用方法
ある	改造できる。	－	○	sharedState で認証情報が渡せるようにして呼び出します。
	改造できない。	sharedState で認証情報が渡せる。	○	jaas.conf（JAAS のコンフィグレーションファイル）をシングルサインオンライブラリ用に変更します。
		sharedState で認証情報が渡せない。	×	－
ない	login 用 API が公開されている。	－	○	ログインモジュールを作成します。
	login 用 API が公開されていない。	－	×	－

（凡例）　○：適用できる　　×：適用できない　　－：該当しない

なお，標準ログインモジュールの WebPasswordLoginModule，WebCertificateLoginModule，WebPasswordLDAPLoginModule および WebPasswordJDBCLoginModule（ユーザ認証を行うログインモジュール）は，すべてシングルサインオンを適用できます。

5.6 カスタムログインモジュールの利用

標準ログインモジュール以外を使用して各アプリケーションのユーザ認証をする場合，カスタムログインモジュールを作成して，標準ログインモジュールと組み合わせて使用できます。

カスタムログインモジュールは，アプリケーション開発者が実装します。

5.6.1 カスタムログインモジュールの概要

カスタムログインモジュールは，標準ログインモジュール以外を使用してアプリケーションのユーザ認証をしたい場合に作成するクラスです。このクラスは，JAAS の SPI である LoginModule インタフェースを継承して作成します。

カスタムログインモジュールは，次に示すディレクトリに格納します。

- Windows の場合
  <Application Server のインストールディレクトリ>¥manager¥modules
- UNIX の場合
  /opt/Cosminexus/manager/modules

なお，カスタムログインモジュールの格納先ディレクトリは，ua.conf（統合ユーザ管理のコンフィグレーションファイル）の，com.cosminexus.admin.auth.custom.modules で変更できます。

カスタムログインモジュールを格納するときの注意事項を次に示します。

- カスタムログインモジュールを格納するディレクトリ内のクラスは，カスタムログインモジュール専用のクラスローダで呼び出されます。そのため，このクラスはアプリケーションで使用できません。アプリケーションで使用する場合は，カスタムログインモジュールを格納するディレクトリを usrconf.cfg（J2EE サーバ用オプション定義ファイル）の add.class.path キーに指定してください
- カスタムログインモジュールは必ず.class ファイルのまま格納してください。JAR ファイル形式にまとめて格納しないでください。また，クラス階層がある場合は，クラス階層構造を保ったままで，カスタムログインモジュールの格納ディレクトリに格納してください。
  例を次に示します。
  （例）カスタムログインモジュールのクラスが「my.login.MyLoginModule」の場合

```
<Application Serverのインストールディレクトリ>
    └manager
      └modules
        └my
          └login
            └MyLoginModule.class
```

- 統合ユーザ管理フレームワークでカスタムログインモジュールを使用する場合，カスタムログインモジュール，およびカスタムログインモジュールに関連するクラスを実行環境に配置しておく必要があり

ます。このときのファイル形式は JAR ファイルには対応していません。バイトコードクラスファイルを配置してください。

5.6.2　カスタムログインモジュールの呼び出し

カスタムログインモジュールは，標準ログインモジュールである DelegationLoginModule または WebSSOLoginModule のどちらかで呼び出します。

- DelegationLoginModule
  シングルサインオンの機能を利用しない場合に使用します。
- WebSSOLoginModule
  シングルサインオンの機能を利用する場合に使用します。

また，カスタムログインモジュールを標準ログインモジュールから呼び出すことで，統合ユーザ管理が管理しているログインセッション管理に参加できます。このセッションは Web コンテナのセッション管理とは別です。統合ユーザ管理のセッションについては，「5.4　統合ユーザ管理のセッション管理」を参照してください。

カスタムログインモジュールは，DelegationLoginModule または WebSSOLoginModule のどちらかのログインモジュールから呼びされます。これらのログインモジュールは，呼び出したカスタムログインモジュールで認証が成功すると自動的に統合ユーザ管理のセッションに参加します。統合ユーザ管理のセッション登録時に必要なユーザ ID は，Subject に関連づけられている Principal オブジェクトから取得します。

シングルサインオン機能を使用する場合は，WebSSOLoginModule を使用します。この場合，2 回目のカスタムログインモジュールの呼び出しでは，最初に統合ユーザ管理のセッションに参加したユーザ ID をキーにシングルサインオン情報リポジトリから必要な認証情報が取得されてから，カスタムログインモジュールに渡されます。リポジトリで暗号化して保存していたものは，復号化してから渡されます。

カスタムログインモジュールの実装方法については，「5.12　カスタムログインモジュールを使用したユーザ認証の実装」を参照してください。

5.7　ユーザ情報の管理

統合ユーザ管理では，ユーザ情報を管理するリポジトリとして，LDAP ディレクトリサーバまたはデータベースを使用できます。この節では，LDAP ディレクトリサーバでのユーザ情報の管理について説明します。

5.7.1　LDAP ディレクトリサーバへのユーザ情報の登録

LDAP ディレクトリサーバを使用する場合，ユーザ情報は次に示す方法で登録できます。

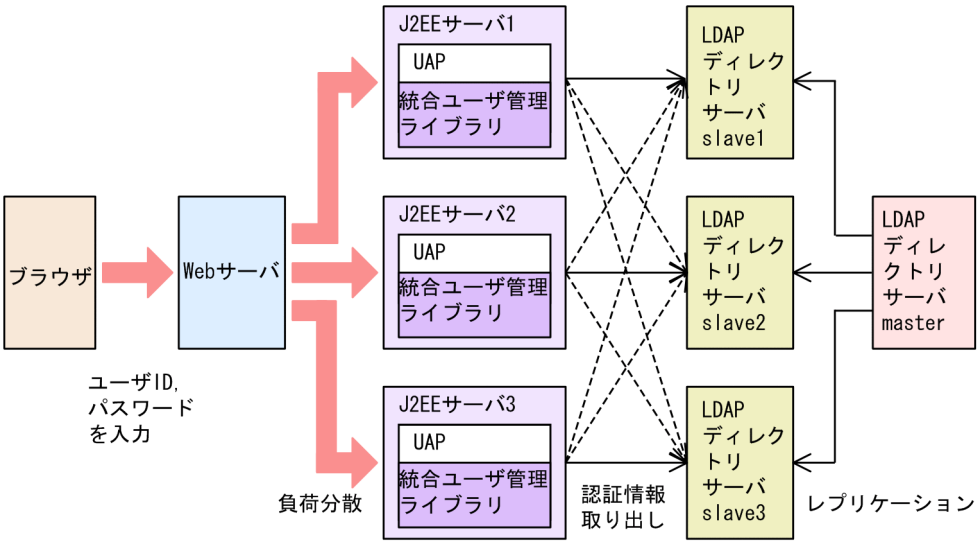
- **LDAP ディレクトリサーバが提供するコマンドによって登録する**
  ご使用の LDAP ディレクトリサーバで提供されているコマンドを使用してください。コマンドによって，ldif ファイルで定義した内容に基づいた一括登録などができます。
- **統合ユーザ管理 API を使用して開発したアプリケーションから登録する**
  ユーザ認証ライブラリおよびシングルサインオンライブラリを使用してユーザ認証を実行するアプリケーションを開発してください。統合ユーザ管理フレームワークを使用したユーザ認証の実装については，「5.9　統合ユーザ管理フレームワークによるユーザ認証の実装」を参照してください。

5.7.2　LDAP ディレクトリサーバの多重化による接続フェイルオーバ

LDAP ディレクトリサーバをレプリケーションによって多重化することで，一つの LDAP ディレクトリサーバへのアクセスに失敗した場合，統合ユーザ管理が提供する標準ログインモジュールはあらかじめ設定した別の LDAP ディレクトリサーバにユーザ情報，シングルサインオン情報の参照先の切り替えを自動的にできます。

例えば，次の図に示す構成では，J2EE サーバ 1 が使用している LDAP ディレクトリサーバ slave1 がダウンした場合，使用する LDAP ディレクトリサーバを自動的に slave2 に切り替えて認証処理ができます（slave2 もダウンしている場合は slave3 へ切り替えます）。

図 5-21　LDAP ディレクトリサーバの多重化の構成例



設定されている LDAP ディレクトリサーバに順番にアクセスし，すべてのサーバでアクセスが失敗した場合，認証が失敗します。

LDAP ディレクトリサーバがダウンしているかどうかの判断方法を次に示します。

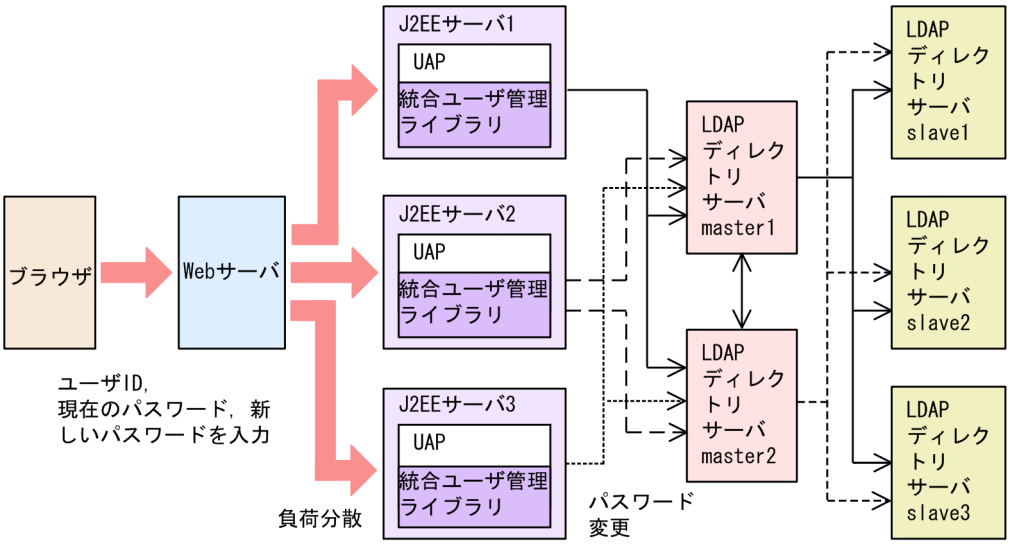
1. javax.naming.CommunicationException 例外が発生します。
   この例外の原因としては，接続先ホストから接続を拒否された場合などが考えられます。詳細については，JDK のドキュメントを参照してください。
2. リトライを実行します。
   リトライは，あらかじめ設定した回数分実行されます。
3. リトライしても LDAP ディレクトリサーバにアクセスできない場合，ダウンしていると見なされます。

設定されているすべての LDAP ディレクトリサーバがダウンしている場合，認証は失敗し，LoginContext クラスの login メソッドの呼び出し元に LoginException 例外が発生します。

各 LDAP ディレクトリサーバへの接続情報は，ua.conf（統合ユーザ管理のコンフィグレーションファイル）で設定して，JAAS アプリケーションごとに jaas.conf（JAAS のコンフィグレーションファイル）で使用する LDAP の設定を一つ以上指定します。ua.conf およびコンフィグレーションファイルの内容の詳細については，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。

また，PasswordUtil クラスによるパスワードの変更も接続フェイルオーバに対応しています。マスタサーバに対してパスワードの変更をします。次の図に示すようなマルチマスタ構成の場合に使用できます。

図 5-22　LDAP ディレクトリサーバの多重化の構成例（マルチマスタ構成）



なお，接続フェイルオーバをする場合，LDAP ディレクトリサーバのエントリのツリー構造およびエントリの内容を同一にしてください。

5.8　統合ユーザ管理フレームワークが提供する API

統合ユーザ管理フレームワークでは，認証処理の呼び出し処理をするアプリケーションを開発するために，JSP のタグライブラリおよび統合ユーザ管理フレームワークのライブラリが提供されています。必要に応じて使用してください。

統合ユーザ管理フレームワークで使用する API の詳細については，「15.　統合ユーザ管理フレームワークで使用する API」を参照してください。

5.8.1　JSP タグライブラリ

統合ユーザ管理フレームワークでは，JSP で統合ユーザ管理フレームワークの機能を容易に利用するための JSP タグライブラリを提供します。アプリケーション開発者（または Web デザイナ）は，Java プログラムによる認証処理の詳細を気にすることなく，容易に JSP を開発できます。

統合ユーザ管理フレームワークのログインモジュールとして標準ログインモジュールを使用する場合，JSP タグライブラリを使用して，ユーザ情報を参照できます。

5.8.2　統合ユーザ管理フレームワークのライブラリ

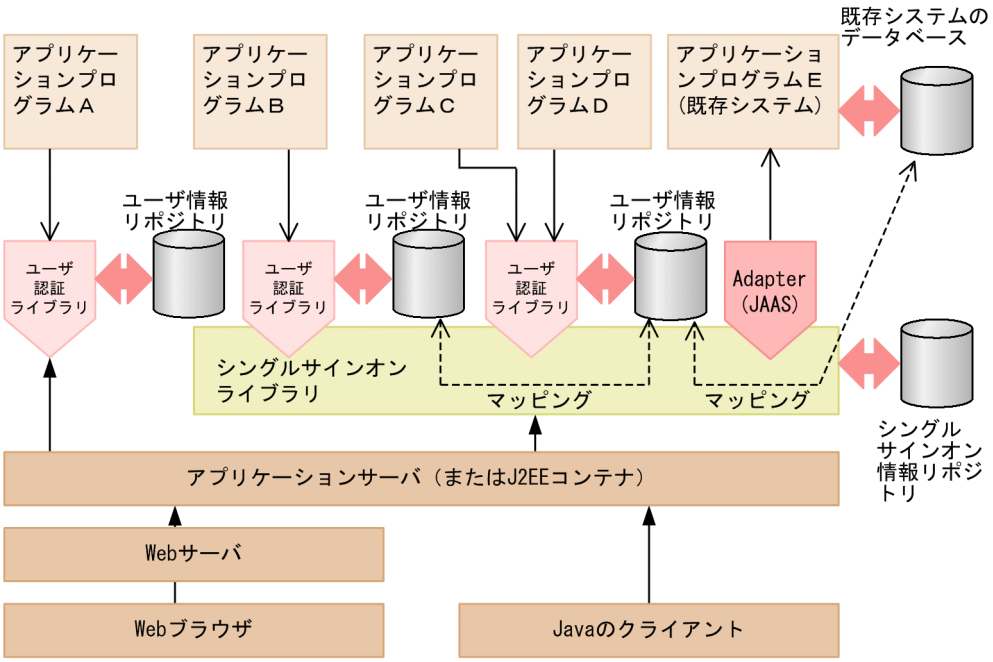
統合ユーザ管理フレームワークは，次の 2 種類で構成されています。

- **ユーザ認証ライブラリ**
  LDAP ディレクトリサーバで構築されたユーザ情報リポジトリの情報を基にユーザ認証をして，認証したユーザの情報をアプリケーションに提供する JAAS 対応ユーザ管理用ライブラリです。
- **シングルサインオンライブラリ**
  ユーザマッピング情報の入ったシングルサインオン情報リポジトリの情報を基にユーザマッピングをして，シングルサインオンを実現するライブラリです。

ユーザ認証ライブラリおよびシングルサインオンライブラリの位置づけを次に示します。

次の図に示すように，ユーザ認証ライブラリは単体でアプリケーションの認証に使用できます。また，シングルサインオンライブラリと協調動作し，マッピングされたユーザの認証をすることもあります。

図 5-23　ユーザ認証ライブラリおよびシングルサインオンライブラリの位置づけ



5.9　統合ユーザ管理フレームワークによるユーザ認証の実装

統合ユーザ管理フレームワークとは，統合ユーザ管理でのユーザ認証を実現するフレームワークです。JAAS に準拠したユーザ認証を，標準ログインモジュールとして提供します。統合ユーザ管理フレームワークを使用することで，独自に認証モジュールを開発しなくても，ユーザ認証を容易に実装できます。

統合ユーザ管理フレームワークによるユーザ認証は，API を使用して実装します。また，ユーザ認証を容易に実装するためのタグライブラリを使用することもできます。API またはタグライブラリを使用してユーザ認証を実装する場合，最初に，サーブレットや JSP でユーザ認証を行うログインモジュールを呼び出します。呼び出したログインモジュールによって認証を行い，処理が終わったところでログアウトします。

標準ログインモジュール以外の認証モジュールを使用する場合は，**カスタムログインモジュール**として作成します。ログインモジュールの種類およびログイン処理の流れについては，「5.2.3　Java 標準仕様（JAAS）に準拠したユーザ認証の概要」，および「5.2.6　統合ユーザ管理の処理の流れ」を参照してください。

ここでは，API とタグライブラリを使用したユーザ認証の実装方法，およびカスタムログインモジュールの作成方法について説明します。

参考

Application Server では，統合ユーザ管理フレームワークの動作確認用のサンプルプログラムを提供します。次に示すファイルに実行手順が記述されていますので，その手順に従って実行してください。

<Application Serverのインストールディレクトリ>¥manager¥examples¥ua¥index.html

5.10　API を使用したユーザ認証の実装

API を使用した場合のログインからログアウトまでの実装方法について説明します。また，次の内容についても説明します。

- ログイン状態のチェック
- パスワード認証時の暗号拡張の実装

なお，統合ユーザ管理フレームワークで提供する API の機能や文法については，マニュアル「アプリケーションサーバ リファレンス API 編」を参照してください。

5.10.1　ログインの実装（API を使用する場合）

統合ユーザ管理フレームワークを使用してユーザ認証をする場合，ログイン時にサーブレットや JSP などでログインモジュールを呼び出す処理を実装します。ログインモジュールを使用するには，JAAS のコンフィグレーションファイルの設定が必要です。JAAS のコンフィグレーションファイルの設定内容については，「14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）」を参照してください。

次に，API を使用した場合の，ログインの実装例を示します。

```
<%@ page import="com.cosminexus.admin.auth.callback.WebPasswordHandler" %>
<%@ page import="javax.security.auth.login.LoginContext" %>
...
<%LoginContext lc = new LoginContext("Portal",
    new WebPasswordHandler(request, response, null, "login.html", true));
  try { lc.login(); } catch (LoginException e) { ... }
%>
...
```

例では，LoginContext クラスからインスタンスを生成し，その引数として，JAAS のコンフィグレーションファイルの Portal エントリに記述された認証モジュールを使用するように指定しています。request に com.cosminexus.admin.auth.name パラメタおよび com.cosminexus.admin.auth.password パラメタが設定されている場合には，これらのパラメタを使用して認証します。そうでない場合は，login.html を呼び出し，ユーザから認証情報（ユーザ ID とパスワード）を取得します。

5.10.2　ユーザ ID の取得の実装（API を使用する場合）

認証が完了したあと，認証したユーザ ID は，Principal オブジェクト（java.security.Principal）として Subject に格納されています。ユーザ ID を取得する場合の実装例を次に示します。

```
<%@ page import="javax.security.auth.Subject" %>
<%@ page import="java.security.Principal" %>
...
<%
```

```
  ...
  Subject subject = lc.getSubject();
  Principal principal = (Principal)subject.getPrincipals().iterator().next();
  String userid = principal.getName();
%>
...
```

Subject から Principal が格納されている iterator を取り出し，その 1 番目に格納されている値を Principal オブジェクトに型変換してから，Principal オブジェクトの getName メソッドでユーザ ID を取得しています。

5.10.3　ユーザ属性の取得の実装（API を使用する場合）

ユーザ属性を取得するには，ログイン時に取得したい属性の一覧を指定する必要があります。ユーザ属性の一覧を指定する場合のログイン処理の実装例を，次に示します。

```
<%@ page import="com.cosminexus.admin.auth.callback.WebPasswordHandler" %>
<%@ page import="com.cosminexus.admin.auth.AttributeEntry" %>
<%@ page import="javax.security.auth.login.LoginContext" %>
...
<%
  AttributeEntry[] attributes = new AttributeEntry[2];
  attributes[0] = new AttributeEntry("cn", "full name", null);
  attributes[1] = new AttributeEntry("employeeNumber", "employee ID", null);
  LoginContext lc = new LoginContext("Portal",
    new WebPasswordHandler(request, response, attributes, "login.html", true));
  try { lc.login(); } catch (LoginException e) { ... }
%>
...
```

属性の一覧に従って属性をリポジトリから取得し，UserAttributes オブジェクトに設定します。取得した属性は，java.lang.Object 型として管理されています。リポジトリから取得した属性を UserAttributes オブジェクトに設定する実装例を次に示します。

```
LoginContext lc = new ...              // LoginContextクラスのインスタンス化
...
Subject subject  = lc.getSubject();
Iterator it = subject.getPublicCredentials().iterator();
UserAttributes ua= (UserAttributes)it.next();   // uaにUserAttributesの
...                                             // リファレンスが格納される。
```

UserAttributes オブジェクト属性の値を参照するには，次のように UserAttributes オブジェクトに対して getAttribute メソッドを使用して String 型で取得します。

```
String role = (String)ua.getAttribute("Portal Role");
```

getAttribute メソッドでユーザ属性を取得する場合の実装例を次に示します。

```
<%@ page import="com.cosminexus.admin.auth.UserAttributes" %>
<%@ page import="javax.security.auth.Subject" %>
...
<%
  ...
  Subject subject = lc.getSubject();
  UserAttributes attrs = (UserAttributes)subject.getPublicCredentials().iterator().next();
  String fullname = (String)attrs.getAttribute("full name");
  String eid = (String)attrs.getAttribute("employee ID");
%>
...
```

5.10.4　認証に成功した Subject を HttpSession に登録する実装

HttpSession に設定するオブジェクトは，java.io.Serializable インタフェースを継承したオブジェクトになります。そのため，HttpSession にはログイン時に生成した LoginContext インスタンスではなく，java.io.Serializable インタフェースを継承している Subject を格納してください。格納した Subject は，ログアウトの実装で必要になります。Subject を HttpSession に格納する実装例（背景色付きの太字部分）を次に示します。

```
<%
  LoginContext lc = new LoginContext("Portal",
    new WebPasswordHandler(request, response, null, "login.html", true));
  try {
    lc.login();
    session.setAttribute("ExampleSubject", lc.getSubject());
  } catch (LoginException e) { ... }
%>
...
```

なお，ログイン後に Subject に関連づけられるユーザ属性（UserAttributes）をセッションフェイルオーバで引き継がせる場合は，Subject とユーザ属性を HttpSession に格納する必要があります。Subject とユーザ属性を HttpSession に格納する実装例（背景色付きの太字部分）を次に示します。

```
<%
  LoginContext lc = new LoginContext("Portal",
    new WebPasswordHandler(request, response, null, "login.html", true));
  try {
    lc.login();
     session.setAttribute("ExampleSubject", lc.getSubject());
     session.setAttribute("ExampleCredential", lc.getSubject().getPublicCredentials().iterat
or().next());
  } catch (LoginException e) { ... }
%>
...
```

5.10.5　ログアウトの実装（API を使用する場合）

ログアウトは「5.10.4　認証に成功した Subject を HttpSession に登録する実装」で HttpSession に登録されている Subject を使用して LoginContext を再生成し，ログアウト処理を実行します。HttpSession に登録されている Subject は削除します。さらに，ユーザ属性を HttpSession に登録している場合はその削除処理も実行します。ユーザ属性がある場合のログアウトの実装例を次に示します。

```
<%
  try {
    Subject subject = (Subject)session.getAttribute("ExampleSubject");
    LoginContext lc = new LoginContext("Example", subject);
    session.removeAttribute("ExampleCredential");
    session.removeAttribute("ExampleSubject");
    lc.logout();
  } catch (LoginException e) { ... }
%>
...
```

セッションタイムアウト時は，HttpSession が無効になるため HttpSession に登録した Subject やユーザ属性などは無効になります。

また，統合ユーザ管理のセッションは，HttpSession と同期しているため無効になります。

5.10.6　ログイン状態のチェック（API を使用する場合）

ログイン状態（ユーザがログインしているかどうかの状態）をチェックするには，Subject オブジェクトが HttpSession に登録されていることと統合ユーザ管理のセッション内でログインしているユーザの有無を調べます。

```
<%
  Subject subject = (Subject)session.getAttribute("mySubject");
  if(subject != null && LoginUtil.check(request, response, realm)){
    // ログインしている場合の処理
  } else {
    // ログインしていない場合の処理
  }
%>
```

5.10.7　パスワード認証時の暗号拡張の実装

統合ユーザ管理フレームワークで標準提供されている，パスワードの暗号の形式（SHA-2，SHA-1，MD5 および平文）以外でも，パスワード認証ができます。この暗号拡張の機能を利用するには，あらかじめ実装クラスを作成する必要があります。

ここでは，暗号拡張できるログインモジュールと，暗号拡張で使用するクラスの実装方法について説明します。暗号拡張の概要については，「5.3.9　パスワード認証時の暗号拡張」を参照してください。

(1) 暗号拡張を利用できるログインモジュール

WebPasswordLoginModule または WebPasswordJDBCLoginModule を使用する場合に，暗号拡張を利用できます。

(2) 暗号拡張で使用するクラスの実装方法

暗号拡張を実装する場合，com.cosminexus.admin.auth.security.PasswordCryptography クラスを継承して処理を実装します。作成したクラスは，次に示すディレクトリの下にクラスファイルで格納します。

- Windows の場合：
  <Application Server のインストールディレクトリ>¥manager¥modules
- UNIX の場合：
  /opt/Cosminexus/manager/modules

なお，格納先ディレクトリは，統合ユーザ管理フレームワークのコンフィグレーションファイル（ua.conf）の，com.cosminexus.admin.auth.custom.modules オプションで変更できます。

SHA-1 形式で byte 型の配列で比較する場合の実装例を次に示します。

```
package my;

import com.cosminexus.admin.auth.security.PasswordCryptography;
import java.security.*;

public class CustomCryptography implements PasswordCryptography
{
  public byte[] encrypt (byte[] plain) {
    byte[] encryptedPassword = null;
    try{
      MessageDigest md = MessageDigest.getInstance("SHA");
      md.update(plain);
      encryptedPassword = md.digest();
    } catch (NoSuchAlgorithmException e) {
      encryptedPassword = plain;
    }
    return encryptedPassword;
  }
}
```

5.10.8 API による実装時の注意事項

ここでは，API を使用したユーザー認証を実装するときの注意事項について説明します。

(1) ログインおよびログアウトを実装するときの注意事項

ログインおよびログアウトを実装するときに，Subject を使用しないで，ログイン時に生成した LoginContext のインスタンスをログアウト時に使用すると，ログインモジュールの設定内容によってはログアウトに失敗するおそれがあります。

ログインおよびログアウトは，Subject を使用する実装にしてください。設定してはいけない実装の例を次に示します。

- ログインおよびログアウトで設定してはいけない実装

```
<%
  LoginContext lc = new LoginContext("Portal",
    new WebPasswordHandler(request, response, null, "login.html", true));
  try { lc.login(); } catch (LoginException e) { ... }
  session.setAttribute("loginContext", lc);
%>
...
<%
  LoginContext lc = (LoginContext)session.getSession().getAttribute("loginContext");
  try { lc.logout(); } catch (LoginException e) { ... }
%>
...
```

注　背景色付きの太字部分は，設定してはいけない実装を示します。

(2) ユーザ情報参照および取得を実装するときの注意事項

ユーザ情報参照および取得を実装するときには，次の点に注意してください。

- UserAttributes オブジェクトの値を変更しても，リポジトリには反映されません。また，ユーザ認証ライブラリでは，取得した属性について何も加工しません。
- UserAttributes オブジェクトに登録されている属性は，String 型だけです。
- 属性の一覧で指定された属性がない場合は，null が設定されます。

5.11　タグライブラリを使用したユーザ認証の実装

タグライブラリを使用した場合のログインからログアウトまでの実装方法について説明します。また，必要なファイルのコピーおよび DD での定義についても説明します。タグライブラリのタグの説明と，タグ属性については，「16.　統合ユーザ管理フレームワークで使用するタグライブラリ」を参照してください。

5.11.1　ログインの実装（タグライブラリを使用する場合）

統合ユーザ管理フレームワークを使用したユーザ認証を行う場合，ログイン時にサーブレットや JSP などでログインモジュールを呼び出す処理を実装します。ログインモジュールを使用するには，JAAS のコンフィグレーションファイルの設定が必要です。JAAS のコンフィグレーションファイルの設定については，「14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）」を参照してください。

JSP の<ua:login/>タグを使用してログインするには，HTTP リクエストオブジェクトに，com.cosminexus.admin.auth.name パラメタ，および com.cosminexus.admin.auth.password パラメタが設定されている必要があります。したがって，まず，次に示すようなログイン用のフォームを準備して，パラメタを設定できるようにします。

```
<html>
<body>
<form action="auth.jsp" method="post">
<table>
<tr>
  <td>username</td>
  <td><input type="text" name="com.cosminexus.admin.auth.name" /></td>
</tr>
<tr>
  <td>password</td>
  <td><input type="password" name="com.cosminexus.admin.auth.password" />
</td>
</tr>
</table>
<br />
<input type="submit" value="Login" />
<input type="reset" value="Reset" />
</form>
</body>
</html>
```

次に，<ua:login/>タグを使用して，JAAS コンフィグレーションファイルの Portal エントリに記述された認証モジュールを使用してログインします。

```
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>
<%@ page errorPage="error.jsp" %>

<ua:login id="lc" entry="Portal" />
...
```

タグライブラリの仕様によって，タグの処理中に発生した例外は，すべて JspException となります。<ua:login/>タグの処理中に発生した例外をより細かく検出する場合は，<ua:exception>Body </ua:exception>タグを使用します。次の例では，発生した例外を，検出する JSP（loginError.jsp）に転送しています。

```
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>

<ua:login id="lc" entry="Portal" excepId="ex" excepScope="session" />
<ua:exception name="ex" ><jsp:forward page="loginError.jsp" /></ua:exception>
...
```

例外を検出する JSP（loginError.jsp）では，例外の内容に応じて返すメッセージを選択しています。

```
<%@ page contentType="text/html; charset=Shift_JIS" %>
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>

<html>
<body>
<ua:exception name="ex" type="javax.security.auth.login.FailedLoginException">
ユーザIDかパスワードが間違っています。<br />
</ua:exception>
<ua:exception name="ex" type="javax.security.auth.login.AccountExpiredException">
アカウントの有効期限が切れています。<br />
</ua:exception>
<ua:exception name="ex" type="javax.security.auth.login.CredentialExpiredException">
パスワードの有効期限が切れています。<br />
</ua:exception>
<ua:exception name="ex" >
例外が発生しました。<br />
<%= ex.toString() %><br />
</ua:exception>
</body>
</html>
```

ポイント

ログインしているかどうかを確認する方法

各 JSP ページの先頭に<ua:notLogin>Body</ua:notLogin>タグを記述することで，その JSP ページを処理する前にログインしているかどうかを確認できます。

```
<%@ page contentType="text/html; charset=Shift_JIS" %>
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>
...
<ua:notLogin>
<a href="login.html">ログインしてください。</a>
</ua:notLogin>
...
```

5.11.2　ユーザ ID の取得の実装（タグライブラリを使用する場合）

認証が完了したあと，認証したユーザ ID は<ua:getPrincipalName>タグを使用して，表示したり，取得したりできます。次に，ユーザ ID を表示する例を示します。

```
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>
...
User ID: <ua:getPrincipalName name="lc" />
...
```

<ua:getPrincipalName>タグの name 属性に，ログイン時に指定した LoginContext の識別子（lc）を指定します。

次に，ユーザ ID を取得する例を示します。

```
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>
...
<ua:getPrincipalName name="lc" id="userid" />User ID: <%= userid %>
...
```

<ua:getPrincipalName>タグの name 属性に加え，id 属性を指定します。id 属性では，ユーザ ID を参照するインスタンスを識別する識別子を指定します。

5.11.3　ユーザ属性の取得の実装（タグライブラリを使用する場合）

ユーザ属性を取得するには，<ua:attributeEntries>タグを使用して取得したい属性の一覧を指定する必要があります。次に，ユーザ属性の一覧を指定する場合の実装例を示します。

```
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>

<ua:attributeEntries id="ae">
  <ua:attributeEntry attrName="cn" alias="full name" />
  ...
</ua:attributeEntries>
<ua:login id="lc" entry="Portal" attrEntName="ae" />
...
Full Name: <ua:getAttribute name="lc" attrName="full name" />
...
```

ユーザ属性の一覧を指定したあとに，<ua:getAttribute>タグを使用して，ユーザ属性を取得します。

```
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>

<ua:login id="lc" entry="Portal" attrFile="MyAttrs.csv" />
...
<ua:getAttribute name="lc" attrName="full name" id="fullname" />
Full Name: <%= fullname %>
...
```

ユーザ情報参照，取得時の注意事項

- UserAttributes の値は参照専用です。内容を変更してもリポジトリには反映されません。また，ユーザ認証ライブラリでは，取得した属性について何も加工しません。
- 登録されている属性は，String 型だけです。
- 属性の一覧で指定された属性がない場合は，null が設定されます。

5.11.4　ログアウトの実装（タグライブラリを使用する場合）

ログアウトの実装例を次に示します。

```
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>

<ua:logout name="lc" />
...
```

<ua:login/>タグに対応する<ua:logout/>タグを明示的に記述しなかった場合は，セッション切断時に暗黙的にログアウトされます。

5.11.5　uatags.jar および uatags.tld のコピーと DD での定義

JSP タグライブラリを使用する場合には，ファイルのコピーや編集が必要です。

JSP タグライブラリ用の JAR ファイル（uatags.jar）とタグライブラリ記述子ファイル（uatags.tld）をコピーして，Web アプリケーションの DD（web.xml）を編集してください。手順を次に示します。

1. uatags.jar を作成する Web アプリケーションの WEB-INF¥lib（Windows の場合），または WEB-INF/lib（UNIX の場合）にコピーします。
2. uatags.tld を作成する Web アプリケーションの WEB-INF にコピーします。
3. web.xml の適切な位置に次の記述を追加します。

```
<taglib>
  <taglib-uri>http://cosminexus.com/admin/auth/uatags</taglib-uri>
  <taglib-location>/WEB-INF/uatags.tld</taglib-location>
</taglib>
```

5.12　カスタムログインモジュールを使用したユーザ認証の実装

標準ログインモジュール以外の手段でユーザ認証する場合は，カスタムログインモジュールとして作成して，標準ログインモジュールと連携して使用します。

5.12.1　標準ログインモジュールと連携するための実装

JAAS では，一度の認証で複数の認証モジュールを順次呼び出せます。これらの認証モジュールでは，LoginModule インタフェースの initialize メソッドの第 3 パラメタに渡される Map オブジェクト（sharedState）を利用して情報を渡します。ここでは，標準ログインモジュール別に，追加する情報を示します。なお，DelegationLoginModule および WebPasswordJDBCLoginModule では，情報は追加しません。

(1)　WebPasswordLoginModule，WebCertificateLoginModule，および WebPasswordLDAPLoginModule

WebPasswordLoginModule，WebCertificateLoginModule，および WebPasswordLDAPLoginModule では，次の LoginModule を呼び出す前に sharedState に次に示す情報を追加します。

キー：com.cosminexus.admin.auth.userattributes

値の型：UserAttributes

説明：Subject に関連づけたユーザ属性を格納した UserAttributes オブジェクトを参照します。

設定されるタイミング：commit メソッド終了直前

(2)　WebSSOLoginModule

同一セッション内で認証済みであれば，そのユーザに対応するユーザマッピングから該当レルムのユーザの認証情報を取得して，カスタムログインモジュールを呼び出す前に次に示す情報を sharedState に追加します。該当セッションで初めて認証する場合や，ユーザマッピング上に認証情報がない場合は，情報を追加しません。

キー：com.cosminexus.admin.auth.sso.userid

値の型：String

説明：ユーザマッピングの USERID で定義した値です。

設定されるタイミング：カスタムログインモジュールの login メソッドを呼び出す直前

キー：com.cosminexus.admin.auth.sso.secdat

値の型：String

説明：ユーザマッピングの SECRETDATA で定義した値です。sharedState に格納される場合は，復号化されて格納されます。

設定されるタイミング：カスタムログインモジュールの login メソッドを呼び出す直前

キー：com.cosminexus.admin.auth.sso.pubdat

値の型：String

説明：ユーザマッピングの PUBLICDATA で定義した値です。

設定されるタイミング：カスタムログインモジュールの login メソッドを呼び出す直前

また，各キーについては，統合ユーザ管理フレームワークのコンフィグレーションファイルで変更できます。すでにカスタムログインモジュールがあり，sharedState から認証情報を取得できる場合，そのカスタムログインモジュールの仕様に合わせたキーにできます。コンフィグレーションファイルの設定内容については，「14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）」を参照してください。

5.12.2　カスタムログインモジュールの実装時の留意点

カスタムログインモジュールを作成するには，JAAS の SPI である LoginModule インタフェースを継承して必要な処理を実装します。カスタムログインモジュールを実装する場合の，LoginModule インタフェースの実装時の留意点を示します。また，ユーザ ID を管理する Principal オブジェクトの実装時の留意点を示します。

(1)　LoginModule インタフェースの実装時の留意点

- login メソッド

  シングルサインオンに対応するには，最初に sharedState 内にユーザ ID とパスワードが指定されていないか判断してください。なお，sharedState からユーザ ID とパスワードを取得するための名前については，統合ユーザ管理フレームワークのコンフィグレーションファイルで指定できます。

- commit メソッド

  Principal オブジェクトを Subject に設定してください。Principal オブジェクトが複数ある場合，WebSSOLoginModule および DelegationLoginModule では最初に見つかった Principal オブジェクトを使用して統合ユーザ管理のセッションへユーザ ID の登録をします。シングルサインオンの場合は，最初にログインしたユーザ ID を認識するために使用します。

- logout メソッド

  logout メソッドは，commit メソッドで Subject に関連づけた Principal オブジェクトや Credential（ユーザ属性など）を削除します。また，ログインで確保したリソースがあれば，確保したリソースを解放します。

  logout メソッド使用時に次の現象が発生することがあります。

  - logout メソッドが呼ばれたとき，Subject に Credential が設定されない
  - ログアウト時にカスタムログインモジュールの commit メソッドまたは login メソッドで設定したメンバ属性の値が参照できない

  Credential が設定されない現象は，シリアライズ化できる Subject オブジェクトのシリアライズ対象に，Credential が含まれていないために発生することがあります。

ログアウト時にメンバ属性の値が参照できない現象は，JAAS の LoginContext（LoginModule を含む）がシリアライズ化されたオブジェクトではないために発生します。LoginContext は HttpSession に Subject オブジェクトを格納し，ログアウトするために Subject オブジェクトから新たなログインモジュールのインスタンスを生成するため，commit メソッドや login メソッドで設定したメンバ属性の値が参照できなくなります。

(2) Principal オブジェクトの実装時の留意点

Principal オブジェクトは，java.security.Principal と java.io.Serializable のインタフェースを継承して実装してください。

5.12.3 カスタムログインモジュールの実装例

カスタムログインモジュールおよび Principal オブジェクトの実装例を示します。

カスタムログインモジュールの実装例では，最初にセッションフェイルオーバ機能を使用しない場合の例を示します。セッションフェイルオーバ機能を使用する場合の例は，セッションフェイルオーバ機能を使用しない場合と異なる部分だけ示します。

(1) セッションフェイルオーバ機能を使用しない場合の実装例

セッションフェイルオーバ機能を使用しない場合のカスタムログインモジュールの実装例を示します。

```
/**
* LoginModuleの実装クラスは，LoginModuleインタフェースを継承して作成します。
*
*/
public class ExampleLoginModule implements LoginModule
{
    // initialize()メソッドに渡ってきたパラメタの値を各メソッドで参照するために使用されま
す。
    private Subject subject;
    private CallbackHandler callbackHandler;
    private Map sharedState;
    private Map options;


    // sharedStateからユーザIDおよびパスワードの値を取り出すときの名称を定義します。
    // “simple.login.username”， ”simple.login.password”;は，統合ユーザ管理のコンフィグ
レーションファイルで
    // 指定できます。
    private static final String USERNAME = ”simple.login.username”;
    private static final String PASSWORD = ”simple.login.password”;


    // 認証の判定をするユーザIDを格納します。この値は，login()で設定され，commit()で参照され
ます。
    private String username;
```

```
    // login()の正否を格納します。trueなら，login()が成功しており，falseならlogin()が失敗し
ています。
    // この値は，login()で設定されcommit()で参照されます。
    private boolean succeeded;


    // commit()の正否を格納します。trueなら，commit()が成功しており，falseならcommit()が失敗
しています。
    // この値は，commit ()で設定されabort ()で参照されます。
    private boolean commitSucceeded;


    /**
    * initialize()メソッドでは，引数に渡ってきたパラメタをメンバ変数に覚えます。
    * また，これ以外に初期化が必要な処理があればそれを行います。
    *（このクラスインスタンス化時に1回だけ呼ばれる）
    *
    */
    public void initialize(Subject subject, CallbackHandler callbackHandler, Map sharedState
, Map options)
    {
        this.subject = subject;
        this.callbackHandler = callbackHandler;
        this.sharedState = sharedState;
        this.options = options;
    }


    /**
    * login ()メソッドでは，認証に必要なユーザIDの取得と認証処理を行います。
    * 本例では，認証に成功した場合succeededにtrueを設定します。また，usernameに認証したユー
ザID
    * を格納します。
    *
    */
    public boolean login()
        throws LoginException
    {
        // SSOに対応するには，最初にsharedState内にユーザID・パスワードを取り出します。
        this.username = (String)this.sharedState.get(USERNAME);
        String password = (String)this.sharedState.get(PASSWORD);


        // sharedState内にユーザIDがない場合，CallbackHandlerを使ってユーザID/パスワードの
        // 取り出しを行います。（この例では，WebPasswordCallbackに値を設定してくれる，WebPass
wordHandlerが
        // 前提になります）
        if (this.username == null || this.username.length() == 0) {
            WebPasswordCallback webpc = new WebPasswordCallback();
            webpc.setOption(WebPasswordCallback.GETPW);
            Callback callbacks[] = new Callback[] { webpc };
            try {
                this.callbackHandler.handle(callbacks);
            }
            catch (Exception ex) {
                // 例外処理を行います。
            }
```

```
            // CallbackからユーザIDとパスワードを取り出します。
            this.username = webpc.getName();
            password = webpc.getPassword();
        }

        // 認証で使用するユーザIDの有無を調べます。もし，ユーザIDがない場合は，例外を上げます。
        if (this.username == null || this.username.length() == 0) {
            throw new FailedLoginException();
        }

        // 各アプリケーションの認証処理を行います。
        // 認証した結果，問題がなければsucceededにtrueを設定します。

        /* ここに認証処理を入れる。 */

        if (!succeeded) {
            throw new FailedLoginException();
        }
        return succeeded;
    }

    /**
    * commit ()メソッドでは，認証したことを示すために，PrincipalオブジェクトをSubjectに関連づけます。
    * (SimplePrincipalは，PrincipalとSerializableのインタフェースを継承して作成されたクラスです。
    *
    */
    public boolean commit()
        throws LoginException
    {
        // PrincipalオブジェクトをSubjectに関連づけます。これにより，統合ユーザ管理で管理している
        // ログインセッション管理に参加したり，SSOの機能に対応することができます。
        this.subject.getPrincipals().add( new SimplePrincipal(this.username) );

        /* ユーザ属性があるなら，それもSubjectに関連づける処理を入れる。 */

        return this.commitSucceeded = true;
    }

    /**
    * abort ()メソッドでは，login()メソッドcommit()メソッドで障害があった場合にリカバリするために
    * 呼び出されます。
    *
    */
    public boolean abort()
        throws LoginException
    {
        if (this.commitSucceeded) {
            // Subjectに関連づけているPrincipalやユーザ属性を解放します。
            // この例ではlogout()メソッドを呼び出して解放します。
            logout();
        }
        return true;
    }
```

```
    /**
    * logout ()メソッドでは，ログアウトする場合に呼び出されます。
    * このメソッドでは，Subjectに関連づけられているPrincipalやユーザ属性を解放するために使用
されます。
    *
    *
    */
    public boolean logout()
        throws LoginException
    {
        // SubjectからPrincipalやユーザ属性を削除する処理を入れます。
        // また，login()メソッドで確保したリソースがあれば，それを解放する処理を追加します。
        return true;
    }
}
```

(2) セッションフェイルオーバ機能を使用する場合の実装例

login メソッドと commit メソッドについては，セッションフェイルオーバ機能を使用しない場合と同じです。ここでは，違いのある logout メソッドの実装方法を示します。

```
/**
 * LoginModuleの実装クラスは，LoginModuleインタフェースを継承して作成します。
 */
public class ExampleLoginModule implements LoginModule
{
  /**
   * logout ()メソッドでは，ログアウトする場合に呼び出されます。
   * このメソッドでは，ログイン時に取得したリソースを解放するために使用されます。
   */
  public boolean logout() throws LoginException
  {
    if (callbackHandler != null) {
        WebLogoutCallback callback = new WebLogoutCallback();
        try {
            callbackHandler.handle(new Callback[]{callback});
        } catch (Exception e) { ... }
        String uid = callback.getUserID();
        HttpSession session = callback.getSession();
        // login()メソッドで確保したリソースがあれば，それを解放する処理を追加します。
        //例えば，グローバルセッションに登録した情報を削除するなど。
    }
    return true;
  }
}
```

logout メソッドでは，ログイン時に取得したリソースを解放します。セッションフェイルオーバ機能を使用する場合，Subject や Principal はフェイルオーバされないことに注意してください。

(3) Principal オブジェクトの実装例

Principal オブジェクトは，java.security.Principal と java.io.Serializable のインタフェースを継承して作成してください。Principal オブジェクトの実装例を示します。

```
import java.security.Principal;
import java.io.Serializable;

/**
* Principalの実装クラスは，PrincipalとSerializableインタフェースを継承して作成します。
*
*/
public class SimplePrincipal implements Principal, Serializable
{
  private String name;

  public SimplePrincipal(String name) {
    if (name == null) throw new NullPointerException();
    this.name = name;
  }
  public String getName() { return name; }
  public String toString() { return getName(); }
  public boolean equals(Object o) {
    if (o == null) return false;
    if (this == o) return true;
    if (!(o instanceof SimplePrincipal)) return false;
    SimplePrincipal rhs = (SimplePrincipal)o;
    if (getName().equals(rhs.getName())) return true;
    return false;
  }
  public int hashCode() { return getName().hashCode(); }
}
```

5.13　統合ユーザ管理機能の設定手順

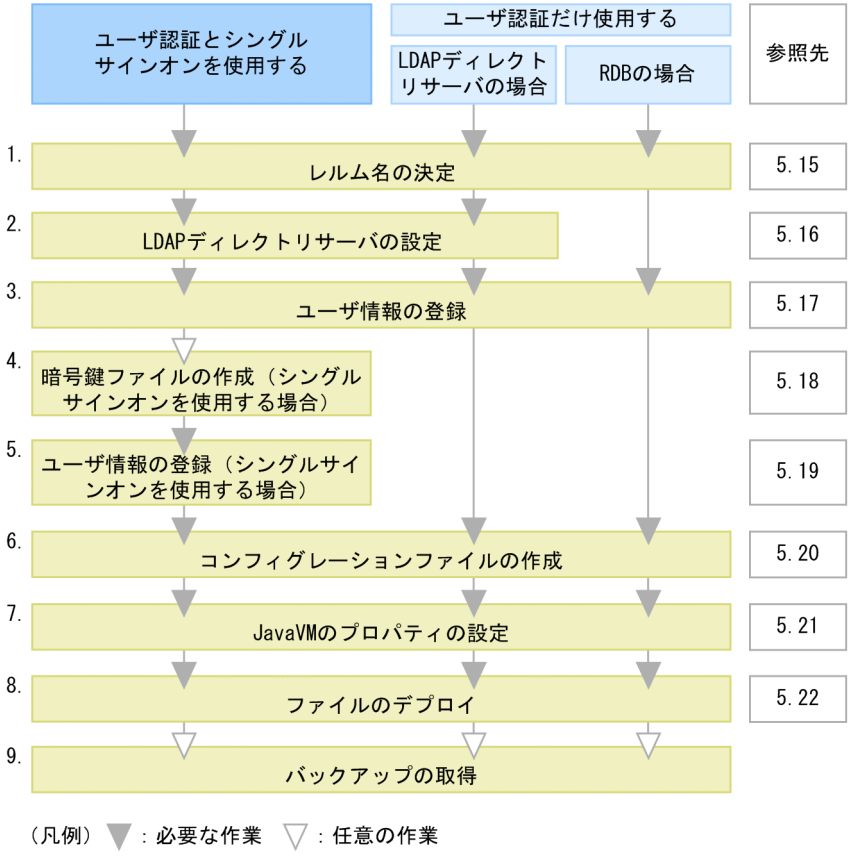
この節では，統合ユーザ管理機能の設定手順について説明します。

Application Server では，Application Server で構築したシステムにログインするユーザを統合管理できます。統合ユーザ管理では，それぞれの J2EE アプリケーションが管理しているユーザの情報が関連づけて管理され，一度のログイン処理でさまざまな J2EE アプリケーションにログインできるようになります。統合ユーザ管理機能を使用する場合は，ユーザ認証情報を格納する LDAP ディレクトリサーバや，統合ユーザ管理のコンフィグレーションファイルを設定する必要があります。

なお，統合ユーザ管理機能を使用するためには，J2EE アプリケーションの開発時に，JAAS の API および Application Server が提供している統合ユーザ管理 API，ならびに JSP タグライブラリを使用して，標準ログインモジュールを呼び出す認証処理プログラムを作成しておく必要があります。また，それぞれのアプリケーションで独自の認証処理をする場合は，カスタムログインモジュールを作成する必要があります。作成方法については，「5.12　カスタムログインモジュールを使用したユーザ認証の実装」を参照してください。

統合ユーザ管理機能の設定手順を次の図に示します。

図 5-24　統合ユーザ管理機能の設定手順



図中の 1.～9.について説明します。

1. ユーザ管理方法を検討し，同一の認証ポリシーを適用する範囲（レルム）を決定します。

ユーザを管理する単位を検討し，対応するレルム名を決定します。レルム名の決定については，「5.14 レルム名の決定」を参照してください。

2. LDAP ディレクトリサーバを設定します。

シングルサインオンを使用する場合には，シングルサインオン用のユーザ情報を LDAP ディレクトリサーバで管理しますので，LDAP ディレクトリサーバが必要です。LDAP ディレクトリサーバの設定については，「5.15　LDAP ディレクトリサーバの設定」を参照してください。

シングルサインオンを使用しないで RDB（HiRDB や Oracle など）の提供するユーザ認証だけを使用する場合は，この手順は不要です。

3. ユーザ認証に使用するユーザ情報を LDAP ディレクトリサーバ，または RDB に登録します。

LDAP ディレクトリサーバへのユーザ情報の登録については，「5.16　ユーザ情報の登録」を参照してください。なお，Application Server では，LDAP ディレクトリサーバに格納するユーザ管理リポジトリの標準的な DIT 構造を規定しています。リポジトリの構造については，「5.2.4　統合ユーザ管理で使用するユーザ情報の管理方法」を参照してください。

RDB へのユーザ情報の登録については，使用している RDB のマニュアルを参照してください。

4. シングルサインオンを使用する場合で，シングルサインオン用のユーザ情報を暗号化したいときは，ユーザ情報を暗号化または復号化するための暗号鍵ファイルを作成します。

暗号鍵ファイルの作成については，「5.17　暗号鍵ファイルの作成（シングルサインオンを使用する場合）」を参照してください。

シングルサインオンを使用しない場合，またはユーザ情報を暗号化しない場合，この手順は不要です。

5. シングルサインオンを使用する場合は，シングルサインオン用のユーザ情報を LDAP ディレクトリサーバに登録します。

LDAP ディレクトリサーバへのシングルサインオン用のユーザ情報の登録については，「5.18　ユーザ情報の登録（シングルサインオンを使用する場合）」を参照してください。なお，Application Server では，LDAP ディレクトリサーバに格納するシングルサインオン用のユーザ管理リポジトリの標準的な DIT 構造を規定しています。リポジトリの構造については，「5.2.4　統合ユーザ管理で使用するユーザ情報の管理方法」を参照してください。

シングルサインオンを使用しない場合，この手順は不要です。

6. コンフィグレーションファイルを作成します。

作成するファイルは，次の 2 種類です。

- jaas.conf（JAAS のコンフィグレーションファイル）
- ua.conf（統合ユーザ管理のコンフィグレーションファイル）

コンフィグレーションファイルの作成については，「5.19　コンフィグレーションファイルの作成」を参照してください。

7. JavaVM のプロパティを設定します。

JavaVM のプロパティの設定については，「5.20　JavaVM のプロパティの設定」を参照してください。

8. 統合ユーザ管理で使用する EAR ファイルをデプロイします。

ファイルのデプロイについては，「5.21　ファイルのデプロイ」を参照してください。

9. 必要に応じて，統合ユーザ管理で使用する情報のバックアップを取ります。

LDAP ディレクトリサーバのリポジトリのバックアップとリストアは，LDAP ディレクトリサービスが提供するコマンド，またはディレクトリゲートウェイで操作します。詳細については，LDAP ディレクトリサーバのマニュアルを参照してください。

jaas.conf，ua.conf，および暗号鍵ファイルのバックアップも取るようにしてください。

5.14　レルム名の決定

ユーザを管理する単位を検討し，対応するレルム名を決定してください。この名称は，JAAS 対応ユーザ管理の認証時に使用します。jaas.conf（JAAS のコンフィグレーションファイル）内の該当するログインモジュールのオプションとして指定します。通常は，「SoumuSystem」のような，ユーザ管理を共有しているアプリケーション群を表すわかりやすい名称を使用します。

5.15　LDAP ディレクトリサーバの設定

この節では，LDAP ディレクトリサーバの設定について説明します。

LDAP ディレクトリサーバの設定手順を次に示します。

1. **LDAP ディレクトリサーバを設置します。**
   詳細については，「5.15.1　LDAP ディレクトリサーバの設置」を参照してください。
2. **LDAP ディレクトリサーバにユーザを登録して，アクセス権を設定します。**
   詳細については，「5.15.2　ユーザの登録とアクセス権の設定」を参照してください。
3. **シングルサインオンを使用する場合は，シングルサインオンライブラリ固有のオブジェクトクラスとユーザ定義属性を LDAP ディレクトリサーバに登録して，オブジェクトクラスとユーザ属性定義を拡張します。**
   詳細については，「5.15.3　オブジェクトクラスとユーザ定義属性の拡張」を参照してください。

5.15.1　LDAP ディレクトリサーバの設置

Application Server では，ユーザ情報を管理するリポジトリとして，LDAP ディレクトリサーバを使用できます。LDAP ディレクトリサーバをインストールして初期設定をしてください。インストールと初期設定については，使用している LDAP ディレクトリサーバのマニュアルを参照してください。

使用できる LDAP ディレクトリサーバの例を次に示します。なお，使用できる LDAP ディレクトリサーバの詳細については「リリースノート」でご確認ください。

使用できる LDAP ディレクトリサーバの例

- Sun Java System Directory Server
- IBM SecureWay Directory（または IBM Directory Server）
- Active Directory

すでに LDAP ディレクトリサーバを使用している場合には，LDAP ディレクトリサーバで使用中のスキーマを参照して，シングルサインオン固有のスキーマが使用されていないことを確認してください。詳細については，「5.15.3　オブジェクトクラスとユーザ定義属性の拡張」を参照してください。

5.15.2　ユーザの登録とアクセス権の設定

統合ユーザ管理で LDAP ディレクトリに接続（バインド）するために，管理用および参照用のユーザを登録してアクセス権を設定してください。

管理用ユーザの DN のことを管理用のバインド DN，参照用ユーザの DN のことを参照用のバインド DN といいます。

管理用のバインド DN は，統合ユーザ管理フレームワークで使用するベースエントリ以下すべてのエントリ，およびエントリに設定されたすべての属性に対して，すべてのアクセス権（Read，Write，Add，Delete，Search，Compare，Selfwrite）を持つバインド DN です。管理用のバインド DN は，LDAP ディレクトリサーバ上のユーザ管理情報を登録，参照，変更，および削除するときに使用します。

参照用のバインド DN は，統合ユーザ管理フレームワークで参照するベースエントリ以下すべてのエントリ，およびエントリに設定されたすべての属性に対して，Read と Search のアクセス権を持つバインド DN です。参照用のバインド DN は，LDAP ディレクトリサーバからユーザ情報などを取得するときに使用します。

5.15.3　オブジェクトクラスとユーザ定義属性の拡張

シングルサインオンを使用する場合は，シングルサインオンライブラリ固有のオブジェクトクラスとユーザ定義属性を LDAP ディレクトリサーバに登録して，オブジェクトクラスとユーザ定義属性を拡張してください。

なお，拡張するオブジェクトクラスとユーザ定義属性の属性はシングルサインオンライブラリ固有のスキーマで，ほかのシステムとは共有できません。すでに LDAP ディレクトリサーバを使用している場合は，LDAP ディレクトリサーバで使用中のスキーマを参照し，シングルサインオン固有のスキーマが使用されていないことを確認してください。

(1)　シングルサインオンライブラリで拡張するオブジェクトクラス

シングルサインオンライブラリ固有のオブジェクトクラスを次の表に示します。

表 5–11　シングルサインオンライブラリ固有のオブジェクトクラス

オブジェクトクラス	OID	必須属性	任意属性
CosminexusSSOEntry	1.2.392.200010.7.6.21	objectClass, CosminexusSSOEntryID, CosminexusSSOUID	CosminexusSSOSecretdata, CosminexusSSOPublicdata, CosminexusSSOMapping

(2)　シングルサインオンライブラリで拡張するユーザ定義属性

シングルサインオンライブラリ固有の属性を次の表に示します。

表 5-12　シングルサインオンライブラリ固有の属性

属性	OID	シンタックス	マルチバリュー／シングルバリュー
CosminexusSSOEntryID	1.2.392.200010.7.4.71	cis	シングルバリュー
CosminexusSSOUID	1.2.392.200010.7.4.72	ces	シングルバリュー
CosminexusSSOSecretdata	1.2.392.200010.7.4.73	bin	シングルバリュー
CosminexusSSOPublicdata	1.2.392.200010.7.4.74	ces	シングルバリュー
CosminexusSSOMapping	1.2.392.200010.7.4.75	dn	マルチバリュー

(3)　拡張するオブジェクトクラスおよびユーザ定義属性の追加手順

拡張するオブジェクトクラスおよびユーザ定義属性の追加手順について，LDAP ディレクトリサーバの種類ごとに説明します。

Sun Java System Directory Server または Oracle Directory Server Enterprise Edition の場合

LDAP ディレクトリサーバが起動していることを確認し，uaschema.slapd.ldif を次のコマンドで LDAP ディレクトリサーバに登録してください。

```
ldapmodify -h <ホスト名> -p <ポート番号> -D <管理用のバインドDN> -w <パスワード> -c -f ua
schema.slapd.ldif
```

IBM SecureWay Directory または IBM Directory Server の場合

LDAP ディレクトリサーバが起動していることを確認し，uaschema.ldif を次のコマンドで LDAP ディレクトリサーバに登録してください。

```
ldapmodify -h <ホスト名> -p <ポート番号> -D <バインドDN> -w <パスワード> -c -f uaschema.l
dif
```

ここで，指定するバインド DN は管理者権限を持っている必要があります。

Active Directory の場合

1. Active Directory でスキーマ修正ができるように設定を変更します。Microsoft 管理コンソール（mmc.exe）を起動して，「スナップインの追加と削除」で Active Directory スキーマを追加してください。「Active Directory スキーマ」を右クリックして「操作マスタ」を選択し，「このドメインコントローラでスキーマの修正が可能」をチェックして［OK］ボタンをクリックしてください。
2. 次のコマンドで uaschema.ad.ldif を Active Directory に登録してください（ログオン先のドメインのドメインコントローラに現在ログオンしているユーザとして接続する場合です）。

```
    ldifde -i -c ”dc=domain” ”<ToDN>” -f uaschema.ad.ldif
```

ここで，ToDN にはドメインによって異なる適切な DN を指定してください。例えば，ドメインが「xxx.co.jp」の場合，ToDN は「dc=xxx,dc=co,dc=jp」となります。

5.16　ユーザ情報の登録

この節では，LDAP ディレクトリサーバへのユーザ情報の登録方法と，登録するユーザ情報の文法について説明します。RDB へのユーザ情報の登録方法については，使用している RDB のマニュアルを参照してください。

なお，Application Server では，LDAP ディレクトリサーバに格納するユーザ管理リポジトリの標準的な DIT 構造を規定しています。リポジトリの構造については，「5.2.4　統合ユーザ管理で使用するユーザ情報の管理方法」を参照してください。

ユーザ情報をユーザ情報リポジトリに登録する方法には，次の 2 種類があります。

- コマンドによる登録
- 統合ユーザ管理フレームワークのライブラリによる登録

それぞれの登録方法について説明します。また，登録するユーザ情報の文法，および Active Directory を使用する場合の設定についても説明します。

5.16.1　コマンドによる登録

Application Server が提供するコマンドと LDAP ディレクトリサーバが提供するコマンドを使用してユーザ情報を登録する場合の手順を次に示します。

1. **ユーザ情報を，LDIF ファイルに記述します。**
2. **convpw コマンドで LDIF ファイル中のパスワードを暗号化します。**
   LDIF ファイル中に指定されたパスワードが暗号化されます。convpw コマンドについては，「convpw（パスワードの暗号化）」を参照してください。
3. **暗号化した LDIF ファイルを，LDAP ディレクトリサーバが提供する ldapmodify コマンドでユーザ情報リポジトリに登録します。**

また，LDAP ディレクトリサーバが提供する GUI でユーザ情報を登録できる場合もあります。詳細については，使用している LDAP ディレクトリサーバのマニュアルを参照してください。

統合ユーザ管理の LDAP サーバに IBM Tivoli Directory Server を使用する場合，サフィックス DN を登録しただけではユーザが登録できません。ユーザ登録で使用する LDIF ファイルの先頭に，例のようなコードを追加して ldapmodify コマンドを実行してください。

（例）サフィックス DN に「o=apsm.com」を追加した場合

```
dn: o=apsm.com
objectclass: top
objectclass: organization
o: apsm.com
```

5.16.2　統合ユーザ管理フレームワークのライブラリによる登録

ユーザ認証ライブラリが提供する API を使用してアプリケーションを作成し，そのアプリケーションを使用してユーザ情報を登録します。

API を利用するアプリケーションは，最初に LdapUserDataManager オブジェクトを生成します。このクラスのコンストラクタには LDAP ディレクトリサーバの定義識別子を指定します。定義識別子は，ua.conf（統合ユーザ管理のコンフィグレーションファイル）で指定します。定義識別子には LDAP ディレクトリサーバの URL，バインド DN，バインド DN のパスワード，ベース DN などが関連づけられています。なお，LdapUserDataManager オブジェクトは定義ごとに一つ生成するようにしてください。

なお，LDAP ディレクトリサーバとして Active Directory を使用して，ユーザの登録，ユーザ情報の更新，およびリポジトリ情報のパスワードの変更をする場合の方法については，「5.16.4　Active Directory を使用する場合の設定」を参照してください。

ユーザ認証の実装については，「5.10　API を使用したユーザ認証の実装」を参照してください。また，統合ユーザ管理フレームワークで使用する API については，「15.　統合ユーザ管理フレームワークで使用する API」を参照してください。

5.16.3　登録するユーザ情報の文法

ユーザ情報は，次の表に示す文法に従って登録してください。

表 5-13　ユーザ情報の文法

情報の種類	意味	文法
ユーザ ID	ユーザの識別子。	英数字列。 長さは 1～512（単位：文字）。
パスワード	ユーザに対応するパスワード。	英数字列および特殊文字。 長さは 0～512（単位：文字）。

注 1　英数字列は，英字（A～Z，a～z）と数字（0～9）の文字の並びを意味します。

注 2　特殊文字は，次の記号を意味します。

（空白）｜!｜"｜#｜$｜%｜&｜'｜(｜)｜*｜+｜,｜-｜.｜/｜:｜;｜<｜=｜>｜?｜@｜[｜¥｜]｜^｜_（アンダースコア）｜`｜{｜}｜｜（垂直バー）｜~

注 3 「文法」欄で明記されていない場合は，英字の大文字と小文字の区別をします。

注 4　ASCII 文字列で指定してください（文法については，プログラムではチェックしていません）。

注 5　パスワードを平文でリポジトリに格納する場合，空文字（""）をパスワードとして使用しないでください。使用した場合，そのユーザはログインできなくなることがあります。パスワードとして空文字を使用したい場合は，SHA-1 などによる暗号化をしてください。

5.16.4　Active Directory を使用する場合の設定

LDAP ディレクトリサーバとして Active Directory を使用する場合の設定方法について説明します。

統合ユーザ管理フレームワークのライブラリを使用して，ユーザ情報リポジトリへのユーザの登録，ユーザ情報の更新（ユーザのパスワード変更を含む）を行う場合，Active Directory 固有の環境設定，および SSL を使用して接続するための証明書の登録が必要です。

LDAP ディレクトリサーバとして Active Directory を使用する場合の設定項目を，次の表に示します。なお，使用するユーザ認証方式によって設定する項目が異なります。

表 5-14　Active Directory を使用する場合の設定項目一覧

設定項目		パスワード認証	パスワード変更および ユーザの追加・変更	クライアント認証 （X509 証明書）
jaas.conf の設定	必要なログインモジュールの指定	○	○	○
	ログインモジュールのオプションの指定	△	○	－
ua.conf の設定	ユーザ識別属性の指定	○	○	○
	DN でユーザ ID として使用する属性名の指定	－	－	○
	パスワード属性の指定	○	○	－
	LDAP ディレクトリサーバの種類の指定	○	○	－
	日本語など 2 バイト文字を含む DN の変換	△	△	△
	URL プロトコルの変更	△	○	△
SSL 接続の設定		△	○	△

（凡例）

○：必ず設定する

△：必要に応じて設定する

－：設定しない（設定しても無視される）

注※　パスワードの設定やユーザの追加・変更を実施する場合を示します。

Active Directory 固有の環境設定の方法，および SSL を使用して接続するための証明書の登録を次に示します。

ポイント

Active Directory で管理するユーザは，ほかの LDAP ディレクトリサーバで管理するユーザと，オブジェクトクラス，属性が大きく異なります。統合ユーザ管理フレームワークのライブラリを使

用する場合には，オブジェクトクラスに「user」を使用します，その「user」を作成する場合には，「cn」，「unicodePwd」，「sAMAccountName」，および「userAccountControl」の属性を指定します。

「sAMAccountName」には，セキュリティアカウントマネージャ（SAM）のアカウント名を指定します。なお，通常，アカウント名はユーザ ID と同じ値を指定します。

「userAccountControl」には，ユーザアカウントのプロパティフラグを指定します。一般ユーザのユーザエントリを作成する場合は「512」を指定します。ただし，Active Directory がインストールされているサーバのセキュリティポリシーで，パスワードの長さが 1 文字以上に設定されている場合，「userAccountControl」に「512」を指定すると，ユーザを作成できません。この場合は，次のどちらかの対処をして，ユーザエントリを作成してください。

- セキュリティポリシーのパスワードの長さを 0 文字以上に変更して，「userAccountControl」に「512」を指定する。
- セキュリティポリシーのパスワードの長さの設定を変更しないで，「userAccountControl」に「544」を指定する。

(1)　jaas.conf の設定

jaas.conf に設定する内容について説明します。

(a)　ログインモジュールの設定

認証方式にパスワード認証を使用する場合は WebPasswordLDAPLoginModule，クライアント証明を使用する場合は WebCertificateLoginModule を設定します。

(b)　ログインモジュールのオプションの指定（パスワード認証を使用する場合）

WebPasswordLDAPLoginModule のオプションに「ldap.w」を設定します。なお，SSO を使用する場合は「sso.ldap.w」を設定します。

(2)　ua.conf の設定

ua.conf に設定する内容について説明します。

(a)　ユーザ識別属性の指定

ユーザ識別属性に，「cn」（フルネームに該当）または「sAMAccountName」を設定します。

設定例を次に示します。

（例 1）「cn」をユーザ識別属性に使用する場合

```
com.cosminexus.admin.auth.ldap.attr.userid.0=cn
```

（例 2）「sAMAccountName」をユーザ識別属性に使用する場合

```
com.cosminexus.admin.auth.ldap.attr.userid.0=sAMAccountName
```

ユーザ識別属性を「sAMAccountName」に設定した場合リポジトリのユーザエントリ（RDN）の検索が必要となります。検索に必要なプロパティとその設定例を次に示します。

（例）

```
java.naming.security.principal.0=cn=Administrator,cn=Users,dc=cosminexus,dc=com
java.naming.security.credentials.0=adminpassword
com.cosminexus.admin.auth.ldap.search.userrdn.0=true
com.cosminexus.admin.auth.ldap.search.scope.0=onelevel
```

なお，「sAMAccountName」をユーザ識別属性に使用する場合は，LdapUserDataManager クラスの addUserData(String uid, UserData userData)メソッドは使用できません。ユーザを追加する場合は addUserData(String uid, UserData userData, String name, String value)メソッドを使用してください。また，属性名の引数（String name）には，「cn」を設定してください。

(b)　DN でユーザ ID として使用する属性名の指定（クライアント証明書を使用する場合）

クライアント証明書を使用する場合は，クライアント証明書に格納されている DN（クライアント証明書の被認証者識別名）を分解したあと，ユーザ ID として使用する属性名を設定します。設定例を次に示します。

（例）

```
com.cosminexus.admin.auth.ldap.certificate.attr.userid.0=cn
```

ここで指定する属性名は，「(a)　ユーザ識別属性の指定」で指定するユーザ識別属性とは異なります。

(c)　パスワード属性の指定（パスワード認証を使用する場合）

パスワード属性に「unicodePwd」を設定します。設定例を次に示します。

（例）

```
com.cosminexus.admin.auth.ldap.attr.password.0=unicodePwd
```

(d)　LDAP ディレクトリサーバの種類の指定（パスワード認証を使用する場合）

接続先の LDAP ディレクトリサーバの種類に「AD」を指定した行を追加します。LDAP 設定番号が「0」の場合の設定例を次に示します。

（例）

```
com.cosminexus.admin.auth.ldap.directory.kind.0=AD
```

(e) URL プロトコルの変更

接続先の LDAP ディレクトリサーバの URL のプロトコルに「ldaps」を設定します。設定例を次に示します。なお，ポート番号は省略できます。

（例）

変更前：`java.naming.provider.url.0=ldap://localhost:389`

変更後：`java.naming.provider.url.0=ldaps://localhost:636`

(3) SSL 接続の設定

J2EE サーバと Active Directory 間の通信で SSL を使用するための証明書を登録します。証明書の登録方法を次に示します。

1. **作成したデジタル証明書を，Active Directory がインストールされているサーバ（LDAP サーバ）に登録します。**

   デジタル証明書の作成，登録方法については，Active Directory のドキュメントを参照してください。

2. **認証局（CA）の証明書を，J2EE サーバに登録します。**

   認証局の証明書の J2EE サーバへの登録は，Developer's Kit for Java に付属する keytool を使用して実行できます。keytool の詳細については，Java 2 SDK, Standard Edition のドキュメントを参照してください。keytool の実行例を次に示します。なお，表記の都合上，複数行にわたっていますが，実際は一行で記述します。

   Windows の場合

   ```
   keytool -import -alias cakey -file C:¥temp¥cacer.cer -trustcacerts -keystore
   ”<Application Serverのインストールディレクトリ>¥jdk¥lib¥security¥cacerts”
   ```

   UNIX の場合

   ```
   /opt/Cosminexus/jdk/bin/keytool -import -alias cakey -file /tmp/cacer.cer
   -trustcacerts -keystore /opt/Cosminexus/jdk/lib/security/cacerts
   ```

なお，keytool で証明書を登録する際に J2EE サーバを起動していた場合は，J2EE サーバを再起動してください。

(4) 注意事項

注意事項を次に示します。

- ua.conf でユーザのパスワードを表す属性値として「unicodePwd」を指定した場合，パスワード形式オプション「password.encrypt」および「password.encrypt.ex」の設定は無効になります。
- Active Directory では，既存ユーザエントリのオブジェクトクラスの変更はサポートしていません。ua.conf に「com.cosminexus.admin.auth.ldap.directory.kind.0=AD」を指定した場合，LdapUserDataManager クラスのコンストラクタで新規にオブジェクトを追加したあとで，既存のユーザエントリを更新しても，ユーザエントリ作成時のオブジェクトクラスが適用されます。

5.17　暗号鍵ファイルの作成（シングルサインオンを使用する場合）

この節では，シングルサインオン用のユーザ情報を暗号化または復号化するための暗号鍵ファイルの作成と変更について説明します。ユーザ情報を暗号化しない場合，暗号鍵ファイルの作成は不要です。

シングルサインオン用のユーザ情報を暗号化する場合は，暗号鍵ファイルで暗号化して LDAP ディレクトリサーバに格納します。また，ユーザ情報を参照するためには，暗号鍵ファイルで復号化して参照します。ユーザ情報を暗号化する場合は，LDAP ディレクトリサーバに登録する前に，暗号鍵ファイルを作成してください。

5.17.1　暗号鍵ファイルの作成

シングルサインオン用のユーザ情報を暗号化または復号化するための暗号鍵ファイルは，ssogenkey コマンドで作成します。なお，作成した暗号鍵ファイルは，システム管理者が安全な場所に保管してください。

また，シングルサインオン用のユーザ情報を暗号化または復号化するためには，ua.conf（統合ユーザ管理のコンフィグレーションファイル）で，使用する暗号エンジンに「JCE」を指定する必要があります。

ssogenkey コマンドについては，「ssogenkey（暗号鍵ファイルの作成）」を参照してください。ua.conf については，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。

5.17.2　暗号鍵ファイルの変更

シングルサインオン用のユーザ情報は登録済みで，暗号鍵ファイルを変更する場合は，次の手順で変更してください。暗号鍵ファイルを変更する手順を次に示します。

1. ssoexport コマンドでシングルサインオン情報リポジトリの内容をすべて取り出します。
2. ssogenkey コマンドを実行して，暗号鍵ファイルを作成します。
3. ssoimport コマンドで，1.で取り出した内容を登録します。

ssoexport コマンドについては，「ssoexport（シングルサインオン情報リポジトリの参照）」を参照してください。ssogenkey コマンドについては，「ssogenkey（暗号鍵ファイルの作成）」を参照してください。ssoimport コマンドについては，「ssoimport（シングルサインオン情報リポジトリの登録）」を参照してください。

5.18　ユーザ情報の登録（シングルサインオンを使用する場合）

この節では，LDAP ディレクトリサーバへのシングルサインオン用のユーザ情報の登録方法と，登録するユーザ情報の文法について説明します。

なお，Application Server では，LDAP ディレクトリサーバに格納するシングルサインオン用のユーザ管理リポジトリの標準的な DIT 構造を規定しています。リポジトリの構造については，「5.2.4　統合ユーザ管理で使用するユーザ情報の管理方法」を参照してください。

シングルサインオン用のユーザ情報をシングルサインオン情報リポジトリに登録する方法には，次の 2 種類があります。

- コマンドによる登録
- 統合ユーザ管理フレームワークのライブラリによる登録

それぞれの登録方法について説明します。また，登録するユーザ情報の文法についても説明します。

5.18.1　コマンドによる登録

シングルサインオン用のユーザ情報を CSV 形式ファイルで作成して，Application Server が提供する ssoimport コマンドを使用してシングルサインオン情報リポジトリに登録します。

ssoimport コマンドについては，「ssoimport（シングルサインオン情報リポジトリの登録）」を参照してください。CSV 形式ファイルについては，「14.3　シングルサインオン用認証情報の CSV 形式ファイル」を参照してください。

5.18.2　統合ユーザ管理フレームワークのライブラリによる登録

シングルサインオンライブラリが提供する API を使用してアプリケーションを作成し，そのアプリケーションを使用してユーザ情報を登録します。

API を利用するアプリケーションは，最初に LdapSSODataManager オブジェクトを生成します。このクラスのコンストラクタにはレルム名を指定します。なお，LdapSSODataManager オブジェクトは定義ごとに一つ生成するようにしてください。

シングルサインオン情報リポジトリの内容を API で更新した場合に，それを通知するシングルサインオン用認証情報リスナクラスを実装すれば，シングルサインオン情報リポジトリの更新に同期して接続先システムのリポジトリの更新もできます。

ユーザ認証の実装については，「5.10　API を使用したユーザ認証の実装」を参照してください。また，統合ユーザ管理フレームワークで使用する API については，「15.　統合ユーザ管理フレームワークで使用する API」を参照してください。

5.18.3　登録するユーザ情報の文法

シングルサインオン用のユーザ情報は，次の表に示す文法に従って登録してください。

表 5-15　シングルサインオン用のユーザ情報の文法

情報の種類	意味	文法
レルム名	ユーザ管理の範囲を示す識別子です。	英数字列。 大文字と小文字を区別しません。 DN 名で使用できる名前を付けてください。
ユーザ ID	ユーザ管理機能を持つアプリケーションの使用者を表すユーザの識別子です。	英数字列。 長さは 1～512（単位：文字）。
SecretData	ユーザ管理機能を持つアプリケーションのユーザ ID に対応したパスワードなどの暗号化が必要な認証情報のことです。ユーザを認証するために使用する値を指定します。ここに指定された値は，暗号化されて保存されます。	英数字列および特殊文字。 長さは 0～512（単位：文字）。
PublicData	ユーザ管理機能を持つアプリケーションが，ユーザの認証を行う際に，ユーザ ID および SecretData 以外で必要な認証情報のことです。ここに指定された値は，暗号化されません。	英数字列および特殊文字。 長さは 0～512（単位：文字）。

注 1　英数字列は，英字（A～Z，a～z）と数字（0～9）の文字の並びを意味します。

注 2　特殊文字は，次の記号を意味します。

（空白）｜!｜"｜#｜$｜%｜&｜'｜(｜)｜*｜+｜,｜-｜.｜/｜:｜;｜<｜=｜>｜?｜@｜[｜¥｜]｜^｜_（アンダースコア）｜`｜{｜}｜｜（垂直バー）｜~

注 3 「文法」欄で明記されていない場合は，英字の大文字と小文字の区別をします。

注 4　ASCII 文字列で指定してください。文法については，プログラムではチェックしていません。

5.19　コンフィグレーションファイルの作成

この節では，次の二つのコンフィグレーションファイルの作成，およびコンフィグレーションファイルの設定例を説明します。

- jaas.conf（JAAS のコンフィグレーションファイル）
- ua.conf（統合ユーザ管理のコンフィグレーションファイル）

5.19.1　jaas.conf の作成

jaas.conf には，ユーザ認証ライブラリまたはシングルサインオンライブラリが使用する情報として，アプリケーション単位に，使用するログインモジュール名，ua.conf で指定したリポジトリ（LDAP ディレクトリサーバまたは RDB）の設定番号などを指定します。

(1)　格納場所

jaas.conf の格納場所を次に示します。

- Windows の場合
  <Application Server のインストールディレクトリ>¥manager¥config¥jaas.conf
- UNIX の場合
  /opt/Cosminexus/manager/config/jaas.conf

この jaas.conf を上書きするか，任意の場所にコピーして使用してください。jaas.conf の格納場所は，JavaVM 起動時のプロパティで指定してください。JavaVM 起動時のプロパティの指定については，「5.20　JavaVM のプロパティの設定」を参照してください。

また，Component Container 管理者がファイルを参照できるように jaas.conf のアクセス権を変更してください。Component Container 管理者の設定については，マニュアル「アプリケーションサーバ システム構築・運用ガイド」の「4.1.4　Component Container 管理者を設定するときの注意事項（UNIX の場合）」を参照してください。

(2)　指定内容

アプリケーション単位に次の情報を指定してください。

ユーザ認証ライブラリを使用してユーザ認証する場合

定義するログインモジュールは WebPasswordLoginModule にしてください。この WebPasswordLoginModule のオプションには，ua.conf で定義しているリポジトリで定義した LDAP 設定番号とレルム名を定義します。

WebPasswordLoginModule とカスタムログインモジュールの連携をする場合，ログインモジュールの定義は DelegationLoginModule にしてください。この DelegationLoginModule のオプションに呼び出すカスタムログインモジュール名を定義します。

シングルサインオンライブラリを使用してユーザ認証する場合

定義するログインモジュールは WebSSOLoginModule にしてください。この WebSSOLoginModule のオプションには，ua.conf で定義しているカスタムログインモジュールの識別子とレルム名を定義します。

なお，LDAP ディレクトリサーバとして Active Directory を使用する場合の jaas.conf の設定については，「5.16.4　Active Directory を使用する場合の設定」を参照してください。

jaas.conf については，「14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）」を参照してください。

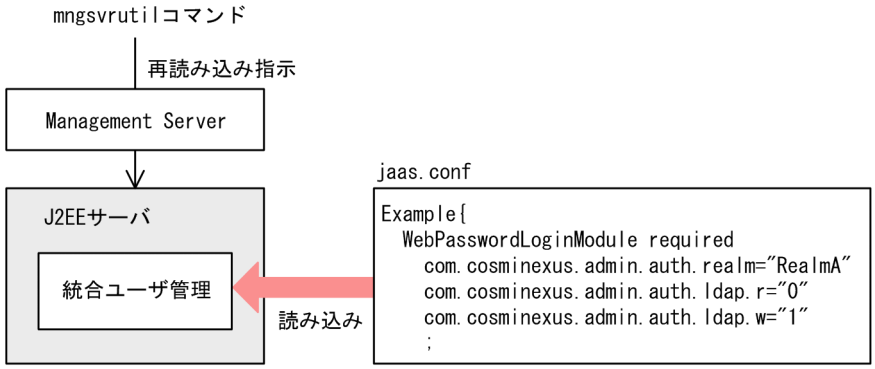
(3)　jaas.conf の再読み込み

Management Server の運用管理コマンドである mngsvrutil コマンドを使用すると，J2EE サーバを再起動しないで，jaas.conf の再読み込みができます。J2EE サーバを再起動しないで，ログインモジュールで使用する LDAP 設定番号を変更したい場合などに利用します。

なお，mngsvrutil コマンドを使用するためには，Management Server が起動されていて，適切に設定済みである必要があります。

jaas.conf の再読み込みの流れを次の図に示します。

図 5-25　jaas.conf の再読み込みの流れ



mngsvrutil コマンドについては，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「mngsvrutil（Management Server の運用管理コマンド）」を参照してください。

5.19.2　ua.conf の作成

ua.conf には，ユーザ認証ライブラリまたはシングルサインオンライブラリが使用する情報として，リポジトリ（LDAP ディレクトリサーバまたは RDB）へのアクセス情報，シングルサインオン用のユーザ情報を暗号化または復号化するための暗号鍵ファイルのパスなどを指定します。

(1)　格納場所

ua.conf の格納場所を次に示します。

- Windows の場合
  <Application Server のインストールディレクトリ>¥manager¥config¥ua.conf
- UNIX の場合
  /opt/Cosminexus/manager/config/ua.conf

この ua.conf を上書きするか，任意の場所にコピーして使用してください。ua.conf の格納場所は，JavaVM 起動時のプロパティで指定します。JavaVM 起動時のプロパティの指定については，「5.20　JavaVM のプロパティの設定」を参照してください。

また，Component Container 管理者がファイルを参照できるように ua.conf のアクセス権を変更してください。Component Container 管理者の設定については，マニュアル「アプリケーションサーバ システム構築・運用ガイド」の「4.1.4　Component Container 管理者を設定するときの注意事項（UNIX の場合）」を参照してください。

(2)　指定内容

ユーザ認証ライブラリの機能を使用したユーザ認証およびシングルサインオンライブラリの機能を使用したシングルサインオンをするために，LDAP ディレクトリサーバの URL やベース DN，アクセス権を設定してください。

シングルサインオンライブラリの機能を使用してシングルサインオンをする場合には，暗号化するための製品の選択，および暗号鍵ファイル名の指定をしてください。また，シングルサインオンの機能から呼び出すログインモジュールがカスタムログインモジュールの場合，カスタムログインモジュール名，およびカスタムログインモジュールと関連するクラスのクラスファイルを格納しているディレクトリを定義してください。

なお，LDAP ディレクトリサーバとして Active Directory を使用する場合の ua.conf の設定については，「5.16.4　Active Directory を使用する場合の設定」を参照してください。

ua.conf の詳細については，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。

(3)　パスワードの変更とスクランブル化

ua.conf に指定した LDAP ディレクトリサーバや RDB に接続するためのパスワードを変更したい場合には，ua.conf を編集するほかに，**uachpw コマンド**を使用して変更できます。uachpw コマンドを使用して LDAP ディレクトリサーバや RDB に接続するためのパスワードを変更する場合に，-scramble オプションを指定すると，パスワードのスクランブル化もできます。

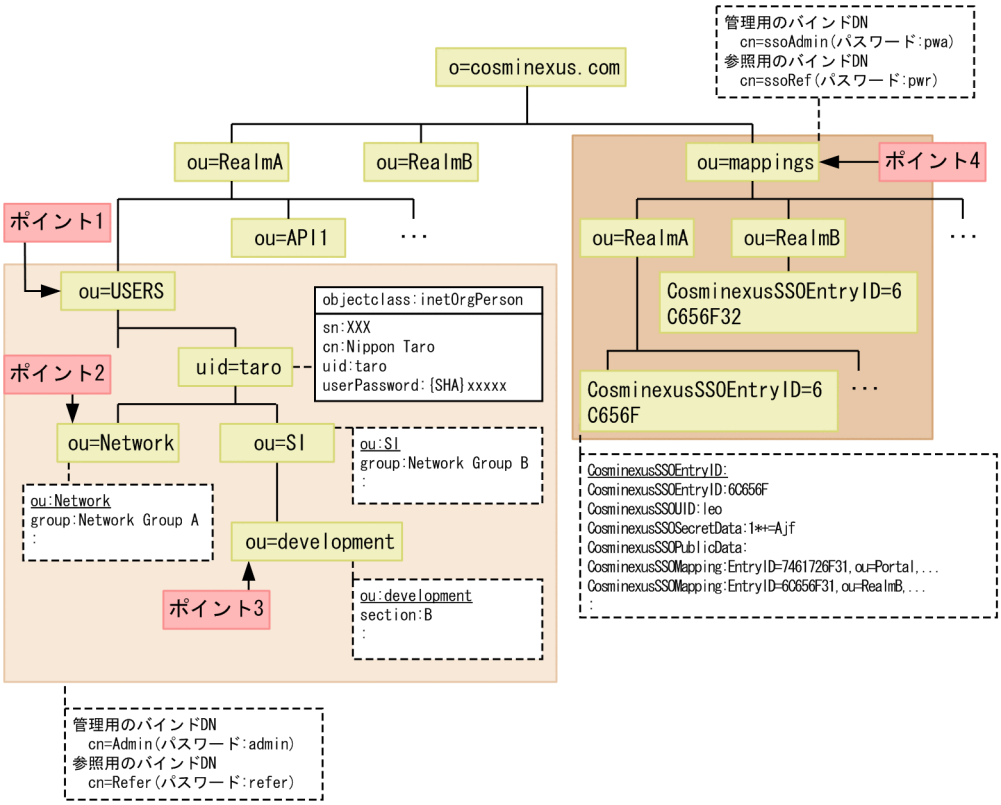
なお，uachpw コマンドでパスワードをスクランブル化する場合には，必ず簡易構築定義ファイルの論理 J2EE サーバ（j2ee-server）の<configuration>タグ内に com.cosminexus.admin.auth.passwordScramble.enable キーを指定して，スクランブル化されたパスワードの復号化機能を有効にしておいてください。

> **参考**
>
> 運用管理ポータルで構築したシステムの場合の説明については，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「3.5　統合ユーザ管理の設定手順」を参照してください。

5.19.3　コンフィグレーションファイルの設定例

ここでは，次の図に示すディレクトリ構成を持つユーザ情報の設定例を示します。

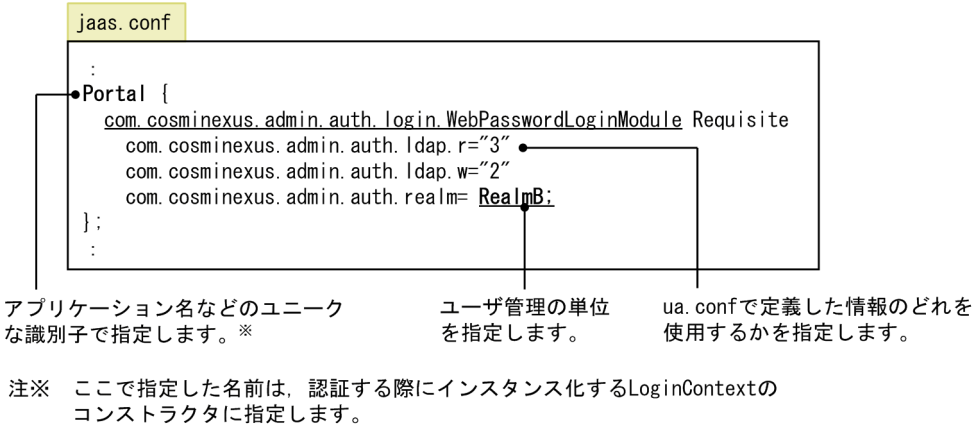
図 5-26　ユーザ情報のディレクトリ構成の例



(1) jaas.conf の設定例

jaas.conf には，ユーザを認証するための情報を設定します。使用するログインモジュール名，ua.conf で指定したリポジトリ（LDAP ディレクトリサーバまたは RDB）の設定番号などを指定します。jaas.conf の設定例を次の図に示します。

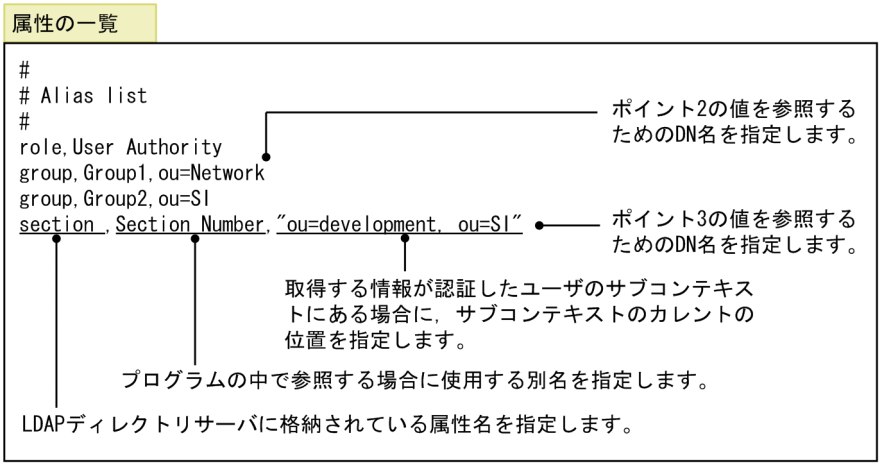
図 5-27　jaas.conf の設定例



(2) 属性の一覧の設定例

属性の一覧には，そのユーザで認証が成功した際に取得するユーザの情報（属性名）と参照に使用する別名（Alias）を指定します。属性の一覧の設定例（c:¥RealmA¥config¥AliasFile.csv（Windows の場合），または/tmp/RealmA/config/AliasFile.csv（UNIX の場合））を次の図に示します。

図 5-28　属性の一覧の設定例



なお，ファイルを作成しなくても，プログラム中で直接 AttributeEntry クラスを使用して指定することもできます。

(3) ua.conf の設定例

ua.conf には，リポジトリ（LDAP ディレクトリサーバまたは RDB）へのアクセス情報などを設定します。ua.conf の設定例を次の図に示します。

図 5-29　ua.conf の設定例（Windows の場合）

ua.conf

```
# シングルサインオン用LDAPサーバへのアクセス定義(管理用)
java.naming.provider.url.0= ldap://localhost:389
java.naming.security.principal.0=cn=ssoAdmin
java.naming.security.credentials.0=pwa
com.cosminexus.admin.auth.ldap.basedn.0= ou=mappings,o=cosminexus.com
com.cosminexus.admin.auth.ldap.pool.enable.0=false

# シングルサインオン用LDAPサーバへのアクセス定義(参照用)
java.naming.provider.url.1= ldap://localhost:389
java.naming.security.principal.1=cn= ssoRef
java.naming.security.credentials.1=pwa
com.cosminexus.admin.auth.ldap.basedn.1= ou=mappings,o=cosminexus.com
com.cosminexus.admin.auth.ldap.pool.enable.1=false

# ユーザ管理用LDAPサーバへのアクセス定義(管理用)
java.naming.provider.url.2= ldap://localhost:389
java.naming.security.principal.2=cn=admin
java.naming.security.credentials.2=admin
com.cosminexus.admin.auth.ldap.basedn.2= ou=USERS,ou=RealmA,o=cosminexus.com
com.cosminexus.admin.auth.ldap.search.userrdn.2=true
com.cosminexus.admin.auth.ldap.search.scope.2=onelevel
com.cosminexus.admin.auth.ldap.attr.userid.2=uid
com.cosminexus.admin.auth.ldap.attr.password.2=userPassword
com.cosminexus.admin.auth.ldap.pool.enable.2=false

# ユーザ管理用LDAPサーバへのアクセス定義(参照用)
java.naming.provider.url.3= ldap://localhost:389
java.naming.security.principal.3=cn=refer
java.naming.security.credentials.3=refer
com.cosminexus.admin.auth.ldap.basedn.3= ou=USERS,ou=RealmA,o=cosminexus.com
com.cosminexus.admin.auth.ldap.search.userrdn.3=true
com.cosminexus.admin.auth.ldap.search.scope.3=onelevel
com.cosminexus.admin.auth.ldap.attr.userid.3=uid
com.cosminexus.admin.auth.ldap.attr.password.3=userPassword
com.cosminexus.admin.auth.ldap.pool.enable.3=false

# WebSSOLoginModuleの定義
com.cosminexus.admin.auth.sso.keyfile=d:/tmp/DES3key.key
com.cosminexus.admin.auth.sso.ldap.r=1
com.cosminexus.admin.auth.sso.ldap.w=0

# Custom LoginModule(Kerberos Version 5)
com.cosminexus.admin.auth.sso.lm.Krb5=test.Krb5LoginModule
com.cosminexus.admin.auth.sso.param.userid.Krb5=javax.security.auth.login.name
com.cosminexus.admin.auth.sso.param.secdat.Krb5=javax.security.auth.login.password

# トレース出力先定義
com.cosminexus.admin.auth.trace.prefix =c:/tmp/LoginTrace
```

アクセス可能なLDAPディレクトリサーバのユーザを指定します。

検索開始点のDNを指定します。この例ではポイント4の値を指定します。

検索開始点のDNを指定します。この例ではポイント1の値を指定します。

ユーザ認証ライブラリでユーザを識別するための属性を指定します。この例では，uidをユーザID，userPasswordをパスワードが格納されている属性として定義しています。

暗号鍵ファイル名を指定します。

トレース情報の出力先を指定します。

図 5-30　ua.conf の設定例（UNIX の場合）

ua.conf

```
# シングルサインオン用LDAPサーバへのアクセス定義(管理用)
java.naming.provider.url.0= ldap://localhost:389
java.naming.security.principal.0=cn=ssoAdmin
java.naming.security.credentials.0=pwa
com.cosminexus.admin.auth.ldap.basedn.0= ou=mappings,o=cosminexus.com
com.cosminexus.admin.auth.ldap.pool.enable.0=false

# シングルサインオン用LDAPサーバへのアクセス定義(参照用)
java.naming.provider.url.1= ldap://localhost:389
java.naming.security.principal.1=cn= ssoRef
java.naming.security.credentials.1=pwa
com.cosminexus.admin.auth.ldap.basedn.1= ou=mappings,o=cosminexus.com
com.cosminexus.admin.auth.ldap.pool.enable.1=false

# ユーザ管理用LDAPサーバへのアクセス定義(管理用)
java.naming.provider.url.2= ldap://localhost:389
java.naming.security.principal.2=cn=admin
java.naming.security.credentials.2=admin
com.cosminexus.admin.auth.ldap.basedn.2= ou=USERS,ou=RealmA,o=cosminexus.com
com.cosminexus.admin.auth.ldap.search.userrdn.2=true
com.cosminexus.admin.auth.ldap.search.scope.2=onelevel
com.cosminexus.admin.auth.ldap.attr.userid.2=uid
com.cosminexus.admin.auth.ldap.attr.password.2=userPassword
com.cosminexus.admin.auth.ldap.pool.enable.2=false

# ユーザ管理用LDAPサーバへのアクセス定義(参照用)
java.naming.provider.url.3= ldap://localhost:389
java.naming.security.principal.3=cn=refer
java.naming.security.credentials.3=refer
com.cosminexus.admin.auth.ldap.basedn.3= ou=USERS,ou=RealmA,o=cosminexus.com
com.cosminexus.admin.auth.ldap.search.userrdn.3=true
com.cosminexus.admin.auth.ldap.search.scope.3=onelevel
com.cosminexus.admin.auth.ldap.attr.userid.3=uid
com.cosminexus.admin.auth.ldap.attr.password.3=userPassword
com.cosminexus.admin.auth.ldap.pool.enable.3=false

# WebSSOLoginModuleの定義
com.cosminexus.admin.auth.sso.keyfile=/opt/Cosminexus/manager/config/DES3key.key
com.cosminexus.admin.auth.sso.ldap.r=1
com.cosminexus.admin.auth.sso.ldap.w=0

# Custom LoginModule(Kerberos Version 5)
com.cosminexus.admin.auth.sso.lm.Krb5=test.Krb5LoginModule
com.cosminexus.admin.auth.sso.param.userid.Krb5=javax.security.auth.login.name
com.cosminexus.admin.auth.sso.param.secdat.Krb5=javax.security.auth.login.password

# トレース出力先定義
com.cosminexus.admin.auth.trace.prefix =/opt/Cosminexus/manager/log/ua
```

アクセス可能なLDAPディレクトリサーバのユーザを指定します。

検索開始点のDNを指定します。この例ではポイント4の値を指定します。

検索開始点のDNを指定します。この例ではポイント1の値を指定します。

ユーザ認証ライブラリでユーザを識別するための属性を指定します。この例では，uidをユーザID，userPasswordをパスワードが格納されている属性として定義しています。

暗号鍵ファイル名を指定します。

トレース情報の出力先を指定します。

認証用プログラムのコーディング例を次の図に示します。

図 5-31　認証用プログラムのコーディング例（Windows の場合）

認証用プログラムのコーディング例

```
 :
String aliasFile ="c:/RealmA/config/AliasFile.csv";
 :
 :
  LoginContext lc = new LoginContext("Portal ",
    new WebPasswordHandler (request, response, aliasFile, "login.html", false));

  try {
    lc .login();
  }
  catch (LoginException e) {
      //エラー処理
  }
 :
Iterator it = subject. getPrincipals().itertor();
String uid = ((java.security.Principal)it.next()).getName();
 :
UserAttributes ua=null;
Iterator it = subject.getPublicCredential().itertor();
while (it.hasNext()) {
  ua = (UserAttributes)it.next();
  if (ua!=null) break;
}  // while
 :
String section = (String)ua .getAttribute("Section_Number");
 :
 :
 :
  try {
    lc .logout();
  }
  catch (LoginException e) {
      //エラー処理
}
```

作成した属性の一覧のファイル名を指定します。

jaas.confで指定した名前を指定します。

認証が成功したときに取得する属性の一覧を指定します。

Principalを取り出します。ユーザIDがわかります。

UserAttributesオブジェクトを取り出します。

UserAttributesオブジェクトから値を取り出します。

aliasFileに記述されているAliasを指定します。

図 5-32　認証用プログラムのコーディング例（UNIX の場合）

認証用プログラムのコーディング例

```
 :
String aliasFile ="/tmp/RealmA/config/AliasFile.csv";
 :
 :
  LoginContext lc = new LoginContext("Portal ",
    new WebPasswordHandler (request, response, aliasFile, "login.html", false));

  try {
    lc .login();
  }
  catch (LoginException e) {
      //エラー処理
  }
 :
Iterator it = subject. getPrincipals().itertor();
String uid = ((java.security.Principal)it.next()).getName();
 :
UserAttributes ua=null;
Iterator it = subject.getPublicCredential().itertor();
while (it.hasNext()) {
  ua = (UserAttributes)it.next();
  if (ua!=null) break;
}  // while
 :
String section = (String)ua .getAttribute("Section_Number");
 :
 :
 :
  try {
    lc .logout();
  }
  catch (LoginException e) {
      //エラー処理
}
```

作成した属性の一覧のファイル名を指定します。

jaas.confで指定した名前を指定します。

認証が成功したときに取得する属性の一覧を指定します。

Principalを取り出します。ユーザIDがわかります。

UserAttributesオブジェクトを取り出します。

UserAttributesオブジェクトから値を取り出します。

aliasFileに記述されているAliasを指定します。

(4)　シングルサインオンへの対応例（標準ログインモジュールの場合）

ユーザ認証ライブラリのログインモジュールをシングルサインオンに対応させるには，jaas.conf を変更する必要があります。シングルサインオンへの対応例（標準ログインモジュールの場合）を次の図に示します。

図 5-33　シングルサインオンへの対応例（標準ログインモジュールの場合）

jaas.conf

```
 :
Portal {
  com.cosminexus.admin.auth.login.WebSSOLoginModule Requisite
    com.cosminexus.admin.auth.ldap.r="1"
    com.cosminexus.admin.auth.ldap.w="0"
    com.cosminexus.admin.auth.realm= RealmB;
};
 :
```

WebSSOLoginModuleに変更することでシングルサインオンに対応できます。

(5)　シングルサインオンへの対応例（カスタムログインモジュールの場合）

カスタムログインモジュールをシングルサインオンに対応させるには，jaas.conf，および認証用プログラムのコーディングを変更する必要があります。また，あらかじめ，ua.conf に，ログインモジュールの識

別子（対応例では「Krb5」）に対応するカスタムログインモジュールの定義項目を定義しておく必要があります。シングルサインオンへの対応例（カスタムログインモジュールの場合）を次の図に示します。

図 5-34　シングルサインオンへの対応例（カスタムログインモジュールの場合）

jaas.conf

```
 :
Portal {
  com.cosminexus.admin.auth.sso.login.WebSSOLoginModule Requisite
    com.cosminexus.admin.auth.sso="Krb5"
    com.cosminexus.admin.auth.realm= RealmB;
    /* 必要に応じて，Krb5LoginModuleのパラメタを定義します。 */
};
 :
```

認証プログラムのコーディング例（変更前）

```
 :
  LoginContext lc = new LoginContext("Portal ",
    new MyCallbackHandler (xxx.yyy.));
 :
```

LoginContextクラスにはWebSSOHandlerを渡し，従来使用していたMyCallbackHandlerはWebSSOHandlerに渡すように変更します。

認証プログラムのコーディング例（変更後）

```
 :
  LoginContext lc = new LoginContext("Portal ",
    new WebSSOHandler(request, responce, new MyCallbackHandler (xxx.yyy.)));
 :
```

5.20　JavaVM のプロパティの設定

統合ユーザ管理機能を使用する場合は，JavaVM の起動時に JavaVM プロパティの設定が必要です。この設定には簡易構築定義ファイルまたは運用管理ポータルを使用します。この節では，SmartComposer 機能を使用して Web システムを構築する場合の簡易構築定義ファイルでの設定方法について説明します。運用管理ポータルでの設定方法については，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「3.5　統合ユーザ管理の設定手順」を参照してください。

JavaVM のプロパティは簡易構築定義ファイルの論理 J2EE サーバ（j2ee-server）の<configuration>タグ内に設定します。

JavaVM のプロパティ内容を，次の表に示します。

表 5-16　簡易構築定義ファイルでの JavaVM のプロパティの設定

指定するパラメタ	設定項目
jaas.ua.enabled	JavaVM の JAAS を有効に設定します。
java.security.auth.login.config	jaas.conf のファイルパスを指定します。
com.cosminexus.admin.auth.config	ua.conf のファイルパスを指定します。
com.cosminexus.admin.auth.passwordScramble.enable	uachpw コマンドでスクランブル化したパスワードの復号化機能の有効または無効を指定します。 詳細については，「5.19.2(3)　パスワードの変更とスクランブル化」を参照してください。
jaas.config.load_exclusively	java.security.auth.login.config パラメタで指定した jaas.conf 以外のログイン構成は無視するかどうかを指定します。

簡易構築定義ファイルについては，マニュアル「アプリケーションサーバ リファレンス 定義編(サーバ定義)」の「4.3　簡易構築定義ファイル」を参照してください。

JavaVM のプロパティ設定例を次に示します。

この例では，パスワードのスクランブル化を有効にし，jaas.conf 以外のログイン構成を無視するように設定をしています。

簡易構築定義ファイルの設定例

```
:
<configuration>
  <logical-server-type>j2ee-server</logical-server-type>
    <param>
      <param-name>jaas.ua.enabled</param-name>
      <param-value>true</param-value>
    </param>
    <param>
      <param-name>java.security.auth.login.config</param-name>
      <param-value><Application Serverのインストールディレクトリ>/manager/config/jaas.con
```

```
f</param-value>
    </param>
    <param>
      <param-name>com.cosminexus.admin.auth.config</param-name>
      <param-value><Application Serverのインストールディレクトリ>/manager/config/ua.conf<
/param-value>
    </param>
    <param>
      <param-name>com.cosminexus.admin.auth.passwordScramble.enable</param-name>
      <param-value>true</param-value>
    </param>
    <param>
      <param-name>jaas.config.load_exclusively</param-name>
      <param-value>true</param-value>
    </param>
:
</configuration>
```

参考

Web アプリケーションで LoginContext クラスを操作するためのアクセス権を変更したい場合は，server.policy の設定を変更してください。

server.policy については，マニュアル「アプリケーションサーバ リファレンス 定義編(サーバ定義)」の「2.2.4　server.policy（J2EE サーバ用セキュリティポリシーファイル)」を参照してください。

5.21　ファイルのデプロイ

この節では，統合ユーザ管理で使用するEARファイルをデプロイ方法について説明します。使用するJ2EEサーバに統合ユーザ管理用のuastartup.earをインポートし，uastartup.earを起動しておいてください。uastartup.earは，次に示すディレクトリにインストールされています。

- Windowsの場合
  <Application Serverのインストールディレクトリ>¥manager¥config
- UNIXの場合
  /opt/Cosminexus/manager/config

uastartup.earは，Management Serverに登録して，J2EEサーバにインポートしてください。J2EEアプリケーションのインポートについては，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)」の「18.5　J2EEアプリケーションのデプロイとアンデプロイ」を参照してください。

統合ユーザ管理フレームワークの機能を使用する場合は，uastartup.earを必ず稼働させてください。稼働していない場合，Management Serverのリソース監視機能，およびmngsvrutilコマンドのサブコマンドreload userAdminは使用できません。

6 アプリケーションの設定による認証

この章では，EJB コンテナや Web コンテナが提供する Web アプリケーションによる認証について説明します。

6.1　この章の構成

この章では，EJB コンテナや Web コンテナが提供する Web アプリケーションによる認証について説明します。

この章の構成を次の表に示します。

表 6–1　この章の構成（アプリケーションの設定による認証）

タイトル	参照先
DD の設定による Web コンテナのユーザ認証	6.2
セキュリティアイデンティティを使用した認証	6.3

6.2　DD の設定による Web コンテナのユーザ認証

Web コンテナは，ロールという仕組みに基づいて認証処理をします。ロールとは，ユーザを管理するための単位で，各ユーザに一つ以上設定されます。ロールは，J2EE アプリケーションに含まれる DD（WEB-INF/web.xml）内の<security-constraint>タグに設定します。J2EE アプリケーションの設定については，「6.2.2　DD での定義」を参照してください。

一方，Web アプリケーションではコンテキスト内の特定の URL ごとに，その URL へアクセスするのに必要なロールを定義できます。認証処理は，Web クライアントが制限された URL へリクエストをする場合に，次に示す二つの段階を踏みます。

- 認証によるリクエストが正当なユーザによるものであるかの判定
- ユーザに設定されたロールと，アクセスするのに必要なロールが一致するかの判定

両方の判定で正当と認められたユーザだけが，制限された URL へアクセスできます。

この節の構成を次の表に示します。

表 6–2　この節の構成（DD の設定による Web コンテナのユーザ認証）

分類	タイトル	参照先
解説	DD の設定による Web コンテナのユーザ認証の機能	6.2.1
実装	DD での定義	6.2.2
設定	実行環境での設定（J2EE アプリケーションの設定）	6.2.3
注意事項	認証機能を併用するときの注意	6.2.4

注　「運用」について，この節での説明はありません。

6.2.1　DD の設定による Web コンテナのユーザ認証の機能

ここでは，DD（WEB-INF/web.xml）の設定による Web コンテナのユーザ認証で実現できる機能について説明します。

(1)　ユーザ情報の管理

Web コンテナでは，各ユーザのユーザ名，パスワード，および所属するロールを，J2EE サーバ付属のユーザ管理機能によって定義し，保持・管理します。

(2)　コンテナセキュリティとアクセス権管理

Web コンテナでは，Web クライアントから特定の URL へのアクセスを制限できます。

特定の URL へのアクセスを制限するには，DD（WEB-INF/web.xml）に次に示す情報を記述します。

- アクセスの制限対象となる URL パターン
- アクセスするのに必要なロールなどのセキュリティ定義情報
- ユーザに定義されたロールを取得するための認証方式

Web クライアントが認証に失敗した場合，またはユーザがアクセスするのに必要なロールを持たない場合には，アクセス制限対象となっている URL パターンへのアクセスはエラーになります。なお，認証に成功したクライアントは，セッションの有効期間内には再度認証の対象になることはありません。

セキュリティ定義情報，および認証方式を定義する方法は，Servlet API 2.3 で規定されている DD（WEB-INF/web.xml）の仕様に従います。

なお，Web コンテナでの認証方式は，Basic 認証と Form 認証とがあります。これらは，J2EE アプリケーションに含まれる DD（WEB-INF/web.xml）に<login-config>タグを追加することで定義できます。J2EE アプリケーションの設定については，「6.2.2　DD での定義」を参照してください。

注意事項

Web サーバに HTTP Server または Microsoft IIS を使用する場合

Web コンテナによる Basic 認証を正しく動作させるには，Web サーバ認証機能を解除しておく必要があります。

詳細については，「6.2.4　認証機能を併用するときの注意」を参照してください。

web.xml で Basic 認証を設定する場合

<realm-name>タグでレルム名を指定してください。省略した場合，レルム名として「Authentication required」が使用されます。なお，<realm-name>タグに空文字列，または空白だけを指定した場合も<realm-name>タグを省略した場合と同じ扱いになります。

(3)　プログラムセキュリティ

DD（WEB-INF/web.xml）によって Basic，または Form 認証によるアクセス制限が設定されたサーブレット，JSP では，HttpServletRequest の次に示す API を使用することで，ログインしたユーザのユーザ名やロール名などに応じて処理を切り替えるなど，プログラムレベルでのより細かいセキュリティ処理を行えます。

- getRemoteUser()
- isUserInRole()
- getUserPrincipal()

これらの API の詳細については，Java Servlet Specification v2.3 を参照してください。

6.2.2　DD での定義

Web コンテナのユーザ認証の設定は，web.xml に指定します。DD での定義を次の表に示します。

表 6-3　DD の設定による Web コンテナのユーザ認証の設定

指定するタグ	設定内容
<security-constraint>タグ	セキュリティ制約を指定します。
<login-config>－<auth-method>タグ	Web コンテナでの認証方法を指定します。

なお，web.xml の設定は，J2EE サーバにデプロイする前の Web アプリケーションで実施します。J2EE サーバにデプロイ済みの Web アプリケーションに設定する場合は属性ファイルに設定します。属性ファイルでの設定については，「6.2.3　実行環境での設定（J2EE アプリケーションの設定）」を参照してください。

6.2.3　実行環境での設定（J2EE アプリケーションの設定）

実行環境での Web コンテナのユーザ認証の設定は，サーバ管理コマンドおよび属性ファイルで実施します。DD の設定による Web コンテナのユーザ認証の設定は，WAR 属性ファイルを使用します。WAR 属性ファイルでの定義を次の表に示します。

表 6-4　DD の設定による Web コンテナのユーザ認証の設定（WAR 属性ファイル）

指定するタグ	設定内容
<security-constraint>タグ	セキュリティ制約を指定します。
<login-config>－<auth-method>タグ	Web コンテナでの認証方法を指定します。

なお，WAR 属性ファイルの設定は，J2EE サーバにデプロイ済みの Web アプリケーションに設定する場合に設定します。J2EE サーバにデプロイする前の Web アプリケーションで設定する場合は，web.xml で設定します。web.xml での設定については，「6.2.2　DD での定義」を参照してください。

6.2.4　認証機能を併用するときの注意

ここでは，Web コンテナの認証機能と，Web サーバの認証機能を併用するときの注意について説明します。

(1)　認証の順序

Web コンテナが提供する認証機能と Web サーバの認証機能を併用した場合，次の順序で認証が実施されます。

1. Web サーバでの認証

2. Web コンテナでの認証

なお，Web サーバの認証機能とは，Web サーバの基本認証，SSL のサーバ認証，SSL のクライアント認証を指します。Web 認証機能の併用とは，Web コンテナが提供する認証機能を使用している場合に，Web サーバの基本認証，SSL のサーバ認証，SSL のクライアント認証のどれか一つ以上を使用していることを指します。

HTTP Server のユーザ認証機能，およびアクセス制御機能については，マニュアル「HTTP Server」を参照してください。

(2) Web サーバの基本認証と Web コンテナの Basic 認証を併用するときの注意

Web サーバの基本認証と Web コンテナの Basic 認証を併用した場合は，Web サーバで認証されたユーザ名とパスワードが Web コンテナに引き継がれます。このため，Web サーバと Web コンテナには，共通のユーザ情報を定義する必要があります。

なお，Web サーバの認証後，Web コンテナでの認証の状況によっては，Web コンテナの動作が異なります。Web コンテナでの動作について次に示します。

- Web サーバで認証されたユーザが，Web コンテナで認証されない場合
  Web コンテナにユーザ名・パスワード入力のダイアログが表示されるので，そこに Web サーバと Web コンテナ共通のユーザ名・パスワードを入力します。
- Web サーバで認証されたユーザが，Web コンテナにアクセスするのに必要なロールを持っていない場合
  アクセス制御対象となっている URL パターンへのアクセスはエラーとなります。
- Web サーバで認証されたユーザが，Web コンテナにアクセスするのに必要なロールを持っている場合
  ユーザ名・パスワードの入力ダイアログが表示されることなく，アクセス制限対象となっている URL パターンへアクセスできます。

(3) Web サーバに Microsoft IIS を使用している場合の注意

Web コンテナの認証を使用する場合，Microsoft IIS で提供する次の認証機能には制限があります。

- ダイジェスト認証
  Web コンテナでの認証機能の使用の有無にかかわらず，ダイジェスト認証は使用できません。Microsoft IIS で，必ずダイジェスト認証の設定を解除してください。
- 統合 Windows 認証
  Web コンテナの Basic 認証を使用する場合は，統合 Windows 認証は使用できません。Microsoft IIS で，統合 Windows 認証の設定を解除してください。

6.3　セキュリティアイデンティティを使用した認証

この節では，セキュリティアイデンティティを使用した認証について説明します。

この節の構成を次の表に示します。

表 6-5　この節の構成（セキュリティアイデンティティを使用した認証）

分類	タイトル	参照先
解説	セキュリティアイデンティティの機能	6.3.1
実装	EJB クライアントアプリケーションでのセキュリティの実装	6.3.2
設定	セキュリティアイデンティティを使用した認証の設定	6.3.3

注　「運用」および「注意事項」について，この節での説明はありません。

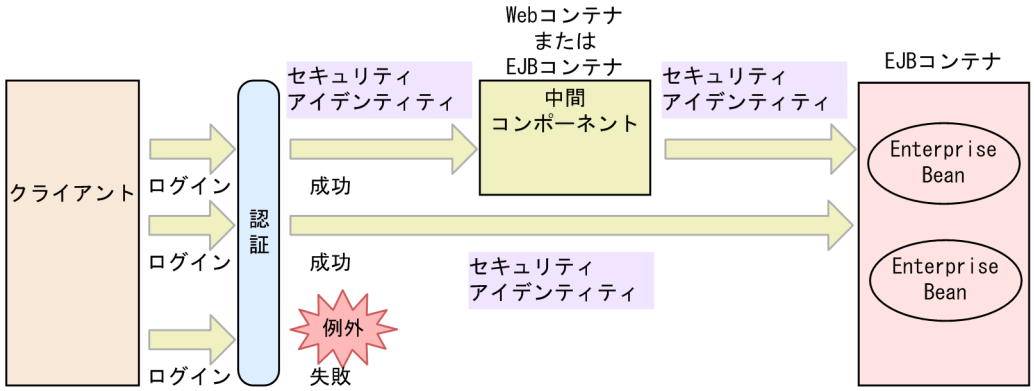
6.3.1　セキュリティアイデンティティの機能

セキュリティ管理機能を使用して，Web コンテナや EJB コンテナへのアクセス時のユーザ認証を実現できます。認証には，ユーザおよびパスワードを使用します。

セキュリティ管理機能での認証に成功した場合は，**セキュリティアイデンティティ**という認証情報が作成され，Web コンテナや EJB コンテナに伝達されます。なお，認証に失敗した場合は例外が発生します。

認証処理でのセキュリティアイデンティティの流れを次の図に示します。

図 6-1　認証処理でのセキュリティアイデンティティの流れ

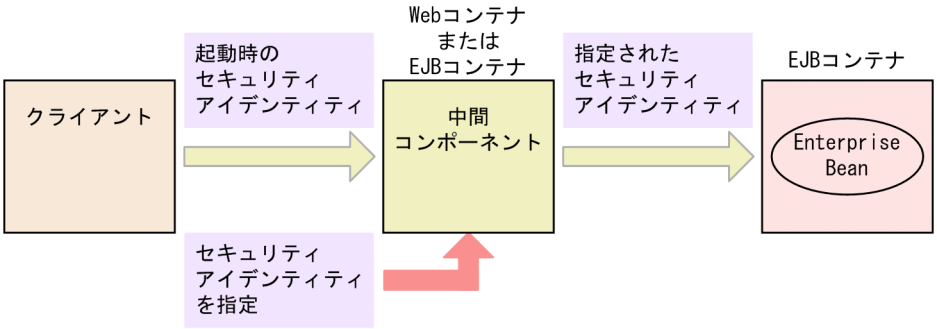


また，セキュリティ管理では，中間コンポーネントで指定したセキュリティアイデンティティを伝達する**Run as 機能**も使用できます。

Run as 機能とは，Enterprise Bean を呼び出す中間コンポーネントで，クライアントでログインしたセキュリティアイデンティティと異なるセキュリティアイデンティティを指定すると，指定したセキュリティ

アイデンティティで Enterprise Bean の呼び出しができる機能です。Run as 機能について次の図に示します。

図 6-2　Run as 機能



なお，アプリケーションサーバのセキュリティ管理機能では，コンポーネントへアクセスするときのメッセージを暗号化する機能，メッセージに署名を付ける機能，証明書による認証機能はサポートしていません。

6.3.2　EJB クライアントアプリケーションでのセキュリティの実装

EJB クライアントアプリケーションでは，J2EE サーバで定義されたユーザとパスワードを使用してユーザを認証できます。EJB クライアントアプリケーションからユーザを認証してログインすると，セキュリティロールが設定された Enterprise Bean のメソッドを呼び出せます。

(1)　実装手順

EJB クライアントアプリケーションでのセキュリティは，Application Server が提供する API を使用して実装します。ここでは，セキュリティを実装する場合の前提条件および実装方法を示します。API の機能と文法については，マニュアル「アプリケーションサーバ リファレンス API 編」の「4.　EJB クライアントアプリケーションで使用する API」を参照してください。

セキュリティを実装する前に，次の前提条件を満たしているか確認してください。

- J2EE サーバ側にユーザが登録されている必要があります。
- 登録されているユーザにセキュリティロールが設定されている必要があります。

EJB クライアントアプリケーションでセキュリティを実装する手順を次に示します。

1. セキュリティ API のパッケージをインポートします。

   セキュリティ API を利用するために，次に示すパッケージをインポートします。

```
import com.hitachi.software.ejb.security.base.authentication.*
```

2. LoginInfoManager のオブジェクトを取得します。

Enterprise Bean のメソッドを呼び出すプログラム上で LoginInfoManager オブジェクトを取得します。取得には LoginInfoManager オブジェクトに用意されているスタティックメソッドの getLoginInfoManager メソッドを使用します。

```
LoginInfoManager  lm = LoginInfoManager.getLoginInfoManager();
```

3. ユーザ名とパスワードでログインします。

LoginInfoManager オブジェクト取得後，login メソッドを呼び出します。

```
lm.login(username, password);
```

4. Enterprise Bean のメソッドを呼び出します。

login メソッド成功後，Enterprise Bean のメソッドを呼び出します。

5. ログアウトします。

Enterprise Bean のメソッド呼び出しが終了したあと，logout メソッドで J2EE サーバからログアウトします。

```
lm.logout();
```

注意事項

EJB クライアントアプリケーションでセキュリティを実装する場合，HiEJBClientStatic.jar をクラスパスに追加してコンパイルする必要があります。

(2) サンプルプログラム

Enterprise Bean 名が account の場合に，getAccountID メソッドを呼び出すサンプルプログラムを次に示します。

```
import com.hitachi.software.ejb.security.base.authentication.*;
    :
  try {
    LoginInfoManager lm = LoginInfoManager.getLoginInfoManager();
    String userName = System.getProperty("username");
    String password = System.getProperty("password");
    if(lm.login(userName , password)) {
      try {
        System.out.println("user:" + userName + "login success");
        Context ctx = new InitialContext();
        java.lang.Object obj = ctx.lookup(appUnitPath + "Account");
        AccountHome aHome =
           (AccountHome)PortableRemoteObject.narrow(obj,AccountHome.class);
        Account account  = aHome.create();
        account.getAccountID();
      } finally {
        lm.logout();
      }
    }
```

```
  } catch(NotFoundServerException e) {
    System.out.println("not found server");
  } catch(InvalidUserNameException e) {
    System.out.println("invalid user name");
  } catch(InvalidPasswordException e) {
    System.out.println("invalid password");
  } catch(Exception e) {
    e.printStackTrace();
  }
```

6.3.3　セキュリティアイデンティティを使用した認証の設定

セキュリティアイデンティティを使用してセキュリティを管理する場合は，サーバ管理コマンドを使用して，ユーザおよびロールの情報を登録します。セキュリティアイデンティティを使用した認証および RunAs 機能を使用するための J2EE アプリケーションの設定を次の表に示します。

表 6-6　セキュリティアイデンティティを使用した認証および RunAs 機能を使用するための J2EE アプリケーションの設定

機能	項目	設定対象	設定内容
セキュリティアイデンティティを使用した認証	セキュリティアイデンティティを使用した認証をするかどうか	Session Bean，Entity Bean，Message-driven Bean	Session Bean 属性ファイル，Entity Bean 属性ファイル，MessageDrivenBean 属性ファイルの<security-identity>タグに，認証するかどうかを指定します。
RunAs 機能	RunAs 機能を使用するかどうか		Session Bean 属性ファイル，Entity Bean 属性ファイル，MessageDrivenBean 属性ファイルの<run-as>タグに，RunAs 機能を使用するかどうかを指定します。
	RunAs 機能で使用するセキュリティロール名		Session Bean 属性ファイル，Entity Bean 属性ファイル，MessageDrivenBean 属性ファイルの<role-name>タグに，セキュリティロール名を指定します。
	RunAs 機能で使用するプリンシパル名		Session Bean 属性ファイル，Entity Bean 属性ファイル，MessageDrivenBean 属性ファイルの<user-name>タグに，プリンシパル名を指定します。

セキュリティアイデンティティの設定手順については，「9.5　セキュリティの定義（セキュリティアイデンティティ）」を参照してください。

7

SSL/TLS 使用による認証情報とデータの暗号化

この章では，SSL/TLS を使用した Web サーバと Web クライアント間のデータの暗号化，および認証について説明します。

7.1　この章の構成

この章では，SSL/TLS 使用によるデータの暗号化と認証の仕組み，および設定方法について説明します。

この章の構成を次の表に示します。

表 7-1　この章の構成（SSL/TLS 使用による認証情報とデータの暗号化）

機能	参照先
SSL 使用による認証情報とデータの暗号化	7.2

7.2　SSL 使用による認証情報とデータの暗号化

SSL をサポートする Web サーバとの連携では，Web サーバと Web クライアント間の SSL による暗号化が有効になります。

この節の構成を次の表に示します。

表 7-2　この節の構成（SSL 使用による認証情報とデータの暗号化）

分類	タイトル	参照先
解説	Web サーバの認証機能	7.2.1
設定	HTTP Server の SSL の設定	7.2.2

注 「運用」および「注意事項」について，この節での説明はありません。

7.2.1　Web サーバの認証機能

Web サーバの機能には，サーバ認証の機能とクライアント認証の機能があります。

サーバ認証

サーバ認証では，サーバにインストールされた鍵交換用の証書を用いて，ブラウザからサーバあてに共通鍵の基となる乱数情報を暗号化して送ります。

この暗号を解くための秘密鍵は，鍵交換用の証書の持ち主であるサーバだけが知っているので，クライアントから見て正当なサーバの場合だけ，ハンドシェイクが成立します。この過程では，サーバは電子署名をしませんが，ハンドシェイクの成立後にサーバが正当であるかどうかの見直しができます。

クライアント認証

クライアント認証では，サーバからブラウザに乱数データを送ってクライアントで電子署名を付与させて，ブラウザにインストールされた電子署名用の証書とともにサーバに送り返させます。

乱数データにブラウザが電子署名を付けて見せることで，ブラウザ自身が秘密鍵を保有していることをサーバに証明します。これによって，サーバ側では，クライアントが証明書に対応した秘密鍵を所有していることを確認できます。

なお，ここで説明する SSL 関連の機能を使用する場合，あらかじめ Web サーバである HTTP Server または Microsoft IIS に SSL の設定をしておく必要があります。HTTP Server の SSL の設定方法については，次を参照してください。

- HTTP Server を使用する場合
  「7.2.2　HTTP Server の SSL の設定」を参照してください。

7.2.2　HTTP Server の SSL の設定

HTTP Server を使用して SSL による認証やデータの暗号化をする場合，秘密鍵の作成，認証局（CA）が発行した証明書の取得，および httpsd.conf（HTTP Server 定義ファイル）の設定が必要です。

また，クライアント認証をする場合，クライアント証明書と認証局（CA）の証明書の取得，および httpsd.conf（HTTP Server 定義ファイル）の設定が必要です。

ここでは，HTTP Server の SSL による認証，暗号化の設定，およびクライアント認証の設定について説明します。なお，詳細については，マニュアル「HTTP Server」の「5.　SSL による認証，暗号化」を参照してください。

8 API による直接接続を使用する負荷分散機の運用管理機能からの制御

アプリケーションサーバの運用管理機能から負荷分散機を制御できます。この章では，API による直接接続を使用する負荷分散機の設定，および運用管理機能から負荷分散機を制御する場合に必要な設定について説明します。

8.1　この章の構成

負荷分散機を使用すると，一つの仮想 IP アドレスで複数のサーバを管理するロードバランシング機能によって，トラフィックを効率的に分散し，処理性能を向上できます。負荷分散機は，アプリケーションサーバの運用管理機能から制御できます。

この章では，API による直接接続の負荷分散機を運用管理機能から制御する場合の設定について説明します。

この章の構成を次の表に示します。

表 8–1　この章の構成（API による直接接続を使用する負荷分散機の運用管理機能からの制御）

分類	タイトル	参照先
解説	API による直接接続を使用する負荷分散機	8.2
	運用管理機能が実行する負荷分散機の API	8.3
設定	負荷分散機の接続環境の設定	8.4
	Management Server（Smart Composer 機能）での負荷分散機の接続情報の設定	8.5
	仮想サーバマネージャでの負荷分散機の接続情報の設定	8.6

注　「実装」,「運用」および「注意事項」について，この章での説明はありません。

8.2　API による直接接続を使用する負荷分散機

アプリケーションサーバで利用できる負荷分散機は，API（SOAP アーキテクチャまたは REST アーキテクチャ）で直接接続します。この場合，事前に設定する必要があります。使用する接続方式ごとに使用できる負荷分散機を次に示します。

API（SOAP アーキテクチャ）による直接接続

- BIG-IP v9
- BIG-IP v10.1
- BIG-IP v10.2
- BIG-IP v11

API（REST アーキテクチャ）による直接接続

- AX2500

8.3　運用管理機能が実行する負荷分散機の API

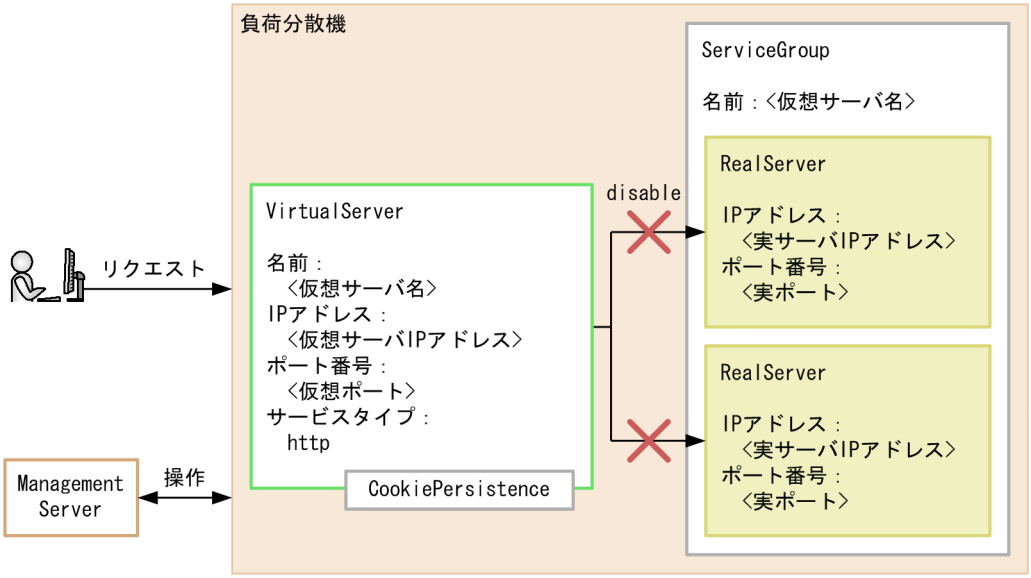
運用管理機能が実行する負荷分散機の API について説明します。

8.3.1　Management Server（Smart Composer 機能）が実行する負荷分散機の API

Smart Composer 機能が実行する負荷分散機の API について説明します。

負荷分散機の API シーケンス図の例を次に示します。ここでは，cmx_build_system（Web システムを構築する場合）で，cookie スイッチングを使用しています。

図 8-1　負荷分散機の API シーケンス図の例（cmx_build_system（Web システムを構築する場合））



VirtualServer は，負荷分散機上で，リクエストの受け口となる仮想的なサーバを表すオブジェクトです。管理ユニットごとに一つ作成されます。ServiceGroup は，負荷分散機上の VirtualServer で受けたリクエストのサービス管理を行うオブジェクトです。VirtualServer ごとに一つ作成されます。RealServer は，負荷分散機上で，VirtualServer で受けたリクエストの振り分け先となる実際のサーバを表すオブジェクトです。管理ユニットに属する仮想サーバの数だけ作成されます。

ACOS で cookie パーシステンスを使用した API で接続する場合は，事前に CookiePersistence（cookie パーシステンス）を作成する必要があります。CookiePersistence では，cookie によるセッションの維持期間を設定します。なお，使用済みの CookiePersistence は必要に応じて削除してください。CookiePersistence の作成，削除方法の詳細は，使用する負荷分散機のドキュメントを参照してください。

参考

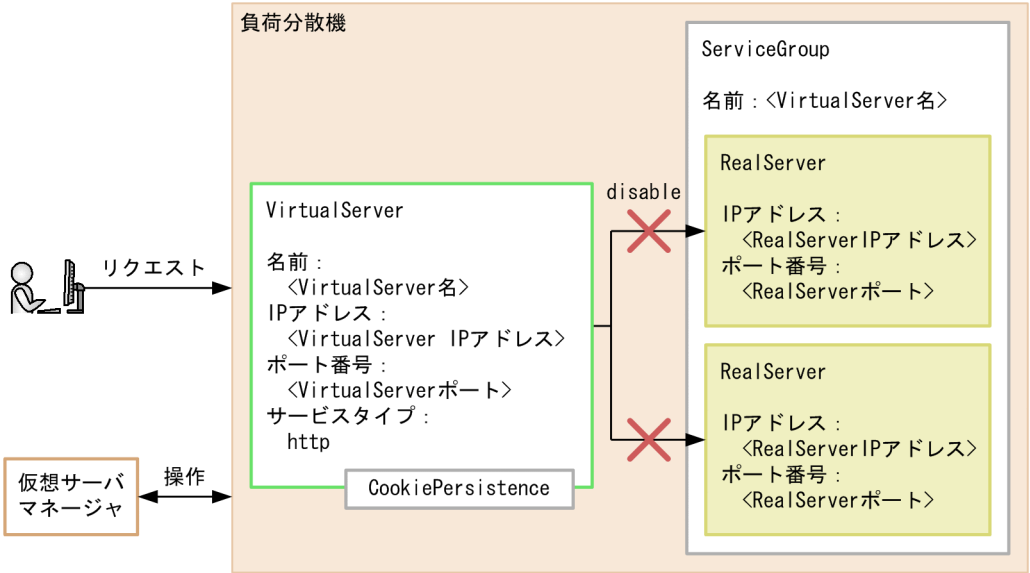
VirtualServer，ServiceGroup，RealServer および CookiePersistence の名称は，負荷分散機の製品によって異なります。

8.3.2　仮想サーバマネージャが実行する負荷分散機の API

仮想サーバマネージャが実行する負荷分散機の API について説明します。

負荷分散機の API シーケンス図の例を次に示します。ここでは，vmiunit update（新規に構築する場合）で，cookie スイッチングを使用しています。

図 8-2　負荷分散機の API シーケンス図の例（vmiunit update（新規に構築する場合））



VirtualServer は，負荷分散機上で，リクエストの受け口となる仮想的なサーバを表すオブジェクトです。管理ユニットごとに一つ作成されます。ServiceGroup は，負荷分散機上の VirtualServer で受けたリクエストのサービス管理をするオブジェクトです。VirtualServer ごとに一つ作成されます。RealServer は，負荷分散機上で，VirtualServer で受けたリクエストの振り分け先となる実際のサーバを表すオブジェクトです。管理ユニットに属する仮想サーバの数だけ作成されます。

ACOS で cookie パーシステンスを使用した API で接続する場合は，事前に CookiePersistence（cookie パーシステンス）を作成する必要があります。CookiePersistence では，cookie によるセッションの維持期間を設定します。なお，使用済みの CookiePersistence は必要に応じて削除してください。CookiePersistence の作成，削除方法の詳細は，使用する負荷分散機のドキュメントを参照してください。

参考

VirtualServer，ServiceGroup，RealServer および CookiePersistence の名称は，負荷分散機の製品によって異なります。

8.4　負荷分散機の接続環境の設定

負荷分散機の接続方式として API による直接接続を使用する場合は，運用管理機能が稼働するホストで負荷分散機の接続環境を設定します。

8.4.1　アクセスリスト（ACL）の設定内容（ACOS の場合）

ACOS のバージョンが「2.4.3-P7」より前の場合は，運用管理サーバ（Management Server）または仮想サーバマネージャが稼働するサーバマシンで，アクセスリストを作成する必要があります。設定内容を次に示します。作成方法の詳細については，ACOS のドキュメントを参照してください。

- ID：「1」
- アクション：「許可」※
- 送信元アドレス：「複数」※

注※

負荷分散機に対して接続制限をする場合は，アクションおよび送信元アドレスに任意の値を設定してください。

注意事項

ID に「1」以外を設定して ACL を作成した場合，API（REST アーキテクチャ）を使用した直接接続が使用できません。

8.4.2　cookie パーシステンステンプレートの作成

cookie を利用してセッションを維持する場合，運用管理機能が稼働するホストで cookie パーシステンステンプレートを作成する必要があります。設定内容を次に示します。なお，作成方法の詳細は，負荷分散機のドキュメントを参照してください。

- cookie 名：「任意の値」
- expire：「0」
  expire に 0 を指定すると，現在のセッションだけが保持されます。

8.4.3　トラストストアの設定

API による直接接続を使用すると，負荷分散機と HTTP または HTTPS で通信します。HTTPS 通信をする場合には，信頼できる証明書を登録したトラストストアが必要になります。HTTPS 通信をする場合は，次のファイルのプロパティで https を指定または省略します。

Management Server で負荷分散機を制御する場合

- lb.properties の lb.API.protocol.<負荷分散機の管理 IP アドレス>

仮想サーバマネージャで負荷分散機を制御する場合

- <LB 接続情報の識別名>.properties の lb.API.protocol
- tierlb.properties の lb.API.protocol

HTTPS 通信をする場合には，あらかじめ，次の作業を実施してトラストストアを設定しておいてください。

1. **負荷分散機から SSL サーバ証明書を取得します。**

   取得方法の詳細は，負荷分散機のドキュメントを参照してください。

2. **運用管理機能が稼働するホストで JDK の keytool コマンドを実行して，手順 1 で取得した SSL サーバ証明書をトラストストアに登録します。**

   JDK の keytool コマンドの実行例を次に示します。

```
<Application Serverのインストールディレクトリ>/jdk/bin/keytool -import -file loadbalancer
.cer -alias loadbalancer -keystore C:¥work¥loadbalancer.keystore -storepass keystore_pass
```

   JDK の keytool コマンドについては，JDK のドキュメントを参照してください。

注意事項

JDK のデフォルトのトラストストア（cacerts）以外に証明書を登録した場合は，lb.properties の javax.net.ssl.trustStore パラメタに SSL サーバ証明書の絶対パスを指定してください。デフォルトのトラストストア（cacerts）に登録した場合は指定する必要はありません。

BIG-IP の場合，デフォルトのトラストストア（cacerts）以外は設定できません。

JDK のデフォルトのトラストストア（cacerts）は，<Application Server のインストールディレクトリ>/jdk/jre/lib/security 下にあります。パスワードの初期値は，「changeit」です。

8.4.4　hosts ファイルの設定（BIG-IP の場合）

Management Server または仮想サーバマネージャから BIG-IP を制御する場合，hosts ファイルに BIG-IP のホスト名と IP アドレスを登録してください。ただし，BIG-IP への接続方法に ssh プロトコルを使用した直接接続を選択した場合は，hosts ファイルへ登録する必要はありません。

8.5 Management Server（Smart Composer 機能）での負荷分散機の接続情報の設定

Smart Composer 機能で負荷分散機の接続情報を定義する場合，Management Server が稼働するホストで，lb.properties（負荷分散機定義プロパティファイル）に接続情報を設定します。

ここでは，API による直接接続を使用する負荷分散機（BIG-IP および AX2500）を設定する場合の設定例を示します。

- BIG-IP（BIG-IP v9，BIG-IP v10.1，BIG-IP v10.2，または BIG-IP v11）の場合

```
lb.list=192.168.100.10

lb.connect_type.192.168.100.10=API
#lb.API.port.192.168.100.10=443
lb.API.user.192.168.100.10=user01
lb.API.passwd.192.168.100.10=user01pw
#lb.API.API.timeout.192.168.100.10=10
```

- AX2500 の場合

```
lb.list=192.168.10.100
lb.enable_passwd.192.168.10.100=adminpw

lb.connect_type.192.168.10.100=API
lb.API.user.192.168.10.100=user01
lb.API.passwd.192.168.10.100=user01pw
#lb.API.port.192.168.10.100=443
#lb.API.cookie_persistence_template.MyWebSystem.192.168.10.100=SC_COOKIE_TEMPNAME
#lb.API.API.timeout.192.168.10.100=10
javax.net.ssl.trustStore=C:¥¥work¥¥ACOS.keystore
javax.net.ssl.trustStorePassword=keystore_pass
```

lb.properties（負荷分散機定義プロパティファイル）については，マニュアル「アプリケーションサーバ リファレンス 定義編(サーバ定義)」の「4.2.4　lb.properties（負荷分散機定義プロパティファイル）」を参照してください。

8.6 仮想サーバマネージャでの負荷分散機の接続情報の設定

管理ユニットの操作時に負荷分散機を利用する場合，使用する負荷分散機の種類，接続方式など，負荷分散機へのアクセスに必要な接続情報を仮想サーバマネージャまたは管理ユニットで定義できます。

8.6.1 負荷分散機の接続情報を仮想サーバマネージャで定義するための設定

仮想サーバマネージャで負荷分散機の接続情報を定義する場合，仮想化システム管理用サーバマシンで，<LB 接続情報の識別名>.properties（負荷分散機接続設定プロパティファイル）に接続情報を設定します。ファイル名の<LB 接続情報の識別名>は，先頭が半角英字から始まる半角英数字，アンダースコア（_）またはハイフン（-）で指定した 31 文字以内の文字列で指定します。

ここでは，API による直接接続を使用する負荷分散機（BIG-IP v9（lb_BIG-IPv9.properties）および AX2500（lb_AX2500.properties））を設定する場合の設定例を示します。

- BIG-IP v9（lb_BIG-IPv9.properties）の場合

```
lb.type=BIG-IPv9
lb.host=192.168.2.14
lb.protocol=API
lb.port=443
lb.user=user01
lb.password=user01pw
lb.timeout=10
```

- AX2500（lb_AX2500.properties）の場合

```
lb.type=ACOS
lb.host=192.168.2.13
lb.protocol=API
lb.port=443
lb.user=user01
lb.password=user01pw
lb.persistence.cookie-insert.templatename=VMI_COOKIE_TEMPNAME
lb.timeout=10
javax.net.ssl.trustStore=C:¥¥work¥¥ACOS.keystore
javax.net.ssl.trustStorePassword=keystore_pass
```

ここで設定した負荷分散機の接続情報を使用する場合は，tier.properties（ティア別プロパティファイル）の lb.use キーに「<LB 接続情報の識別名>」を指定します。設定例の場合，BIG-IP v9 利用時は「lb_BIG-IPv9」，AX2500 利用時は「lb_AX2500」と指定します。

8.6.2 負荷分散機の接続情報を管理ユニットで定義するための設定

管理ユニットで負荷分散機の接続情報を定義する場合，システム構築者は，管理ユニットで管理するティアごとに，tierlb.properties（負荷分散機接続設定プロパティファイル）で接続情報を設定します。

ここでは，API による直接接続を使用する負荷分散機（BIG-IP v9）を設定する場合の設定例を示します。

```
lb.type=BIG-IPv9
lb.host=192.168.2.14
lb.protocol=API
lb.port=443
lb.user=user01
lb.password=user01pw
lb.timeout=10
```

ここで設定した負荷分散機の接続情報を使用する場合は，tier.properties（ティア別プロパティファイル）の lb.use キーに「:unit:」を指定します。

9 サーバ管理コマンドによるセキュリティロールとアプリケーションの操作

この章では，サーバ管理コマンドを使用したセキュリティロールの設定，および J2EE アプリケーションのセキュリティの設定方法について説明します。

9.1　この章の構成

この章では，サーバ管理コマンドを使用したセキュリティロールの設定，および J2EE アプリケーションのセキュリティの設定方法について説明します。

この章の構成を次の表に示します。

表 9–1　この章の構成（サーバ管理コマンドによるセキュリティロールとアプリケーションの操作）

分類	タイトル	参照先
設定	セキュリティロールの設定	9.2
	セキュリティロールのリファレンス定義	9.3
	セキュリティの定義（メソッドパーミッション）	9.4
	セキュリティの定義（セキュリティアイデンティティ）	9.5

注　「解説」,「実装」,「運用」および「注意事項」について，この章での説明はありません。

9.2　セキュリティロールの設定

セキュリティロールを使用したユーザ管理をする場合に必要な設定です。

設定したユーザおよびロールの情報は，J2EE サーバごとに管理されます。

9.2.1　ユーザの設定

ユーザを設定します。

次に示すコマンドを実行して，J2EE サーバにユーザを登録します。

実行形式

```
cjaddsec ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -type user -name <ユーザ名
> -password <パスワード>
```

実行例

```
cjaddsec MyServer -type user -name aps_m -password tiger
```

cjaddsec コマンドの詳細については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「cjaddsec（ユーザとロールの追加）」を参照してください。

9.2.2　ロールの設定

ロールを設定して，ユーザと関連づけます。また，Enterprise Bean およびサーブレットと JSP に参照するセキュリティロールを設定します。

(1)　ロールの登録

次に示すコマンドを実行して，J2EE サーバにロールを登録します。

実行形式

```
cjaddsec ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -type role -name <ロール名
>
```

実行例

```
cjaddsec MyServer -type role -name manage
```

cjaddsec コマンドの詳細については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「cjaddsec（ユーザとロールの追加）」を参照してください。

(2) ロールにユーザを登録

次に示すコマンドを実行してロールにユーザを追加します。

実行形式

```
cjmapsec ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -role <ロール名> -user <
ユーザ名> ［-user <ユーザ名>］
```

実行例

```
cjmapsec MyServer -role manager -user aps_m
```

cjmapsec コマンドの詳細については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「cjmapsec（ユーザとロールのマッピング)」を参照してください。

(3) Enterprise Bean へのセキュリティロールの設定

Enterprise Bean へのセキュリティロールの設定を定義します。

(a) 編集する属性ファイル

EJB-JAR 属性ファイル

(b) 編集する属性ファイルの取得と属性の設定

- 属性ファイルの取得

  次に示すコマンドを実行して EJB-JAR 属性ファイルを取得します。

  実行形式

  ```
  cjgetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリ
  ケーション名> -type ejb -resname <EJB-JAR表示名> -c <EJB-JAR属性ファイルパス>
  ```

  実行例

  ```
  cjgetappprop MyServer -name adder -type ejb -resname adder -c C:¥home¥adder_ejb.xml
  ```

- 属性の設定

  次に示すコマンドを実行して，EJB-JAR 属性ファイルの値を反映します。

  実行形式

  ```
  cjsetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリ
  ケーション名> -type ejb -resname <EJB-JAR表示名> -c <EJB-JAR属性ファイルパス>
  ```

  実行例

  ```
  cjsetappprop MyServer -name adder -type ejb -resname adder -c C:¥home¥adder_ejb.xml
  ```

(c)　編集する属性設定項目

Enterprise Bean のセキュリティロール（<security-role>）の設定項目を次に示します。

項目	必須	対応するタグ名
説明	△	<description>
ロール名	○	<role-name>
セキュリティロール名	△	<linked-to>

（凡例）　○：必須　△：任意

プロパティの設定項目については，マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.3.1　EJB-JAR 属性ファイルの指定内容」を参照してください。

(4)　サーブレットおよび JSP へのセキュリティロールの設定

サーブレットおよび JSP へのセキュリティロールの設定を定義します。

(a)　編集する属性ファイル

WAR 属性ファイル

(b)　編集する属性ファイルの取得と属性の設定

- 属性ファイルの取得

  次に示すコマンドを実行して WAR 属性ファイルを取得します。

  実行形式

```
cjgetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリ
ケーション名> -type war -resname <WAR表示名> -c <WAR属性ファイルパス>
```

  実行例

```
cjgetappprop MyServer -name adder -type war -resname adder -c C:¥home¥adder_war.xml
```

- 属性の設定

  次に示すコマンドを実行して，WAR 属性ファイルの値を反映します。

  実行形式

```
cjsetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリ
ケーション名> -type war -resname <WAR表示名> -c <WAR属性ファイルパス>
```

  実行例

```
cjsetappprop MyServer -name adder -type war -resname adder -c C:¥home¥adder_war.xml
```

(c)　編集する属性設定項目

Web アプリケーション（サーブレットおよび JSP）のセキュリティロールのリファレンス（<security-role>）の設定項目を次に示します。

項目	必須	対応するタグ名
説明	△	<description>
ロール名	○	<role-name>
セキュリティロール名	△	<linked-to>

（凡例）　○：必須　△：任意

プロパティの設定項目については，マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.7.1　WAR 属性ファイルの指定内容」を参照してください。

9.3　セキュリティロールのリファレンス定義

J2EE アプリケーションに含まれる Enterprise Bean および WAR の，一つまたは複数のメソッドに対するセキュリティロールチェックの参照を定義します。このセキュリティチェックは，コンテナで提供されるセキュリティサービスとは別のものです。

9.3.1　Enterprise Bean のセキュリティロールリファレンスの定義

Enterprise Bean のセキュリティロールのリファレンスを定義します。

(1)　編集する属性ファイル

Enterprise Bean の種類に対応する属性ファイルを編集します。

- Session Bean 属性ファイル
- Entity Bean 属性ファイル

(2)　編集する属性ファイルの取得と属性の設定

- 属性ファイルの取得

  次のコマンドを実行して，Enterprise Bean 属性ファイルを取得します。

  実行形式

  ```
  cjgetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type ejb -resname <EJB-JAR表示名>/<Enterprise Bean表示名> -c <Enterprise Beanの属性ファイルパス>
  ```

  実行例

  ```
  cjgetappprop MyServer -name adder -type ejb -resname adder/adder_eb -c C:¥home¥adder_ejb.xml
  ```

- 属性の設定

  次のコマンドを実行して，Enterprise Bean 属性ファイルの値を反映します。

  実行形式

  ```
  cjsetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type ejb -resname <EJB-JAR表示名>/<Enterprise Bean表示名> -c <Enterprise Beanの属性ファイルパス>
  ```

  実行例

  ```
  cjsetappprop MyServer -name adder -type ejb -resname adder/adder_eb -c C:¥home¥adder_ejb.xml
  ```

(3) 編集する属性設定項目

Enterprise Bean のセキュリティロールリファレンス（<security-role-ref>）の設定項目を次に示します。

項目	必須	対応するタグ名
説明	△	<description>
セキュリティロール参照名	○	<role-name>
リンク先のセキュリティロール名※	△	<role-link>

（凡例） ○：必須　△：任意

注※　設定したロール名を指定します。ロール名の設定については，「9.2.2　ロールの設定」を参照してください。また，<role-link>を設定後，EJB-JAR 属性ファイルの設定を行うと<role-link>の値がクリアされますので，再度，セキュリティロールリファレンスを設定してください。

プロパティの設定項目については，次の個所を参照してください。

- マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.4.1　Session Bean 属性ファイルの指定内容」
- マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.5.1　Entity Bean 属性ファイルの指定内容」

9.3.2 サーブレットと JSP のセキュリティロールリファレンスの定義

Web アプリケーション（サーブレットおよび JSP）のセキュリティロールのリファレンスを定義します。

(1) 編集する属性ファイル

サーブレット属性ファイル

(2) 編集する属性ファイルの取得と属性の設定

- **属性ファイルの取得**

  次に示すコマンドを実行してサーブレット属性ファイルを取得します。

  実行形式

```
cjgetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type war -resname <WAR表示名>/<サーブレットおよびJSPの表示名> -c <サーブレット属性ファイルパス>
```

  実行例

```
cjgetappprop MyServer -name adder -type war -resname adder/adder_sv -c C:¥home¥adder_war.xml
```

- **属性の設定**

次に示すコマンドを実行して，WAR 属性ファイルの値を反映します。

実行形式

```
cjsetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type war -resname <WAR表示名>/<サーブレットおよびJSPの表示名> -c <サーブレット属性ファイルパス>
```

実行例

```
cjsetappprop MyServer -name adder -type war -resname adder/adder_sv -c C:¥home¥adder_war.xml
```

(3) 編集する属性設定項目

Web アプリケーション（サーブレットおよび JSP）のセキュリティロールのリファレンス（<security-role-ref>）の設定項目を次に示します。

項目	必須	対応するタグ名
説明	△	<description>
セキュリティロール参照名	○	<role-name>
リンク先のセキュリティロール名※	△	<role-link>

（凡例） ○：必須 △：任意

注※ 設定したロール名を指定します。ロール名の設定については，「9.2.2 ロールの設定」を参照してください。

プロパティの設定項目については，マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.9.1 サーブレット属性ファイルの指定内容」を参照してください。

9.4　セキュリティの定義（メソッドパーミッション）

9.4.1　Enterprise Bean のメソッドパーミッション

メソッドパーミッションの設定方法について説明します。メソッドパーミッションの定義では，セキュリティロールによるアクセス制御を定義します。すべてのユーザにアクセス権限を与えることや，どのユーザにもアクセス権限を与えないこともできます。

メソッドパーミッションを設定できるメソッドは次のとおりです。

- Session Bean
  - ホームインタフェースの create メソッド
  - コンポーネントインタフェースのビジネスメソッド，remove メソッド
- Entity Bean
  - ホームインタフェースの create メソッド，finder メソッド，home メソッド
  - コンポーネントインタフェースのビジネスメソッド，remove メソッド

なお，次に示すメソッドにパーミッションを設定しても使用されません。これらのメソッドのアクセス権限チェックには，コンポーネントインタフェースの remove メソッドに設定したメソッドパーミッションが使用されます。

- javax.ejb.EJBHome の remove(javax.ejb.Handle handle)メソッド
- javax.ejb.EJBHome の remove(Object primaryKey)メソッド
- javax.ejb.EJBLocalHome の remove(Object primaryKey)メソッド

注意事項

CTM を使用するアプリケーションで<Enable Scheduling>プロパティが指定された Stateless Session Bean では，ホームインタフェースの create メソッドにセキュリティロールによるアクセス権限を設定しないでください。設定した場合，デプロイに失敗します。

(1)　編集する属性ファイル

次の Enterprise Bean の種類に対応する属性ファイルを編集します。

- Session Bean 属性ファイル
- Entity Bean 属性ファイル

(2)　編集する属性ファイルの取得と属性の設定

- 属性ファイルの取得

次に示すコマンドを実行して Enterprise Bean の属性ファイルを取得します。

実行形式

```
cjgetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type ejb -resname <EJB-JAR表示名>/<Enterprise Bean表示名> -c <Enterprise Beanの属性ファイルパス>
```

実行例

```
cjgetappprop MyServer -name adder -type ejb -resname adder/adder-eb -c C:¥home¥adder_ejb.xml
```

- **属性の設定**

  次に示すコマンドを実行して，Enterprise Bean の属性ファイルの値を反映します。

実行形式

```
cjsetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type ejb -resname <EJB-JAR表示名>/<Enterprise Bean表示名> -c <Enterprise Beanの属性ファイルパス>
```

実行例

```
cjsetappprop MyServer -name adder -type ejb -resname adder/adder-eb -c C:¥home¥adder_ejb.xml
```

(3) 編集する属性設定項目

セキュリティの定義（メソッドパーミッション）の設定項目（<method-permission>）を次に示します。

項目	必須	対応するタグ名
説明	△	<description>
ロール名	△※	<role-name>
メソッド認証有無	△※	<unchecked>
メソッドの説明	△	<method> - <description>
インタフェース種別	△	<method> - <intf>
メソッド名	△	<method> - <name>

（凡例） △：任意

注　セキュリティの定義（メソッドパーミッション）の設定項目（<method-permission>）は，アノテーションで設定している場合，変更できません。

注※　セキュリティ管理をする場合，ロール名とメソッド認証有無は，次のように，どちらか一つ設定します。

- セキュリティロールによるアクセス権限を設定する場合

  ロール名（<role-name>）を指定します。

- すべてのユーザにアクセス権限を与える場合

  メソッド認証有無（<unchecked>）を指定します。

どのユーザにもアクセス権限を与えない場合は，<method-permission>で指定するのではなく，<exclude-list>の下の<method>にアクセス権を与えないメソッドの情報を指定します。

プロパティの設定項目については，次の個所を参照してください。

- マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.4.1　Session Bean 属性ファイルの指定内容」
- マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.5.1　Entity Bean 属性ファイルの指定内容」

9.5　セキュリティの定義（セキュリティアイデンティティ）

セキュリティアイデンティティの設定には，次の二つの場合があります。

- Enterprise Bean が使用する実行時アイデンティティ情報を設定します。
- サーブレットが使用する実行時アイデンティティ情報を設定します。

9.5.1　Enterprise Bean のセキュリティアイデンティティ

Enterprise Bean のセキュリティアイデンティティを定義します。

セキュリティアイデンティティとしては，「UseCallerIdentity」と「RunAs」の二つのタイプを設定できます。

- UseCallerIdentity
  メソッド実行時に呼び出し元セキュリティアイデンティティを使用し，動作します。
  Enterprise Bean のホームインタフェースやコンポーネントインタフェースのメソッド実行時に，その実行スレッドに関連づけるセキュリティアイデンティティを設定します。
- RunAs
  Role name で指定したロールのアイデンティティに従い，動作します。

(1)　編集する属性ファイル

次の Enterprise Bean の種類に対応する属性ファイルを編集します。

- Session Bean 属性ファイル
- Entity Bean 属性ファイル
- Message-driven Bean 属性ファイル

(2)　編集する属性ファイルの取得と属性の設定

- 属性ファイルの取得
  次に示すコマンドを実行して Enterprise Bean の属性ファイルを取得します。

  実行形式

```
cjgetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリ
ケーション名> -type ejb -resname <EJB-JAR表示名>/<Enterprise Bean表示名>-c <Ente
rprise Bean属性ファイルパス>
```

  実行例

```
cjgetappprop MyServer -name adder -type ejb -resname addr/adder_eb -c C:¥home¥adder_ej
b.xml
```

- **属性の設定**

  次に示すコマンドを実行して，Enterprise Bean の属性ファイルの値を反映します。

  実行形式

```
cjsetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type ejb -resname <EJB-JAR表示名>/<Enterprise Bean表示名> -c <Enterprise Bean属性ファイルパス>
```

  実行例

```
cjsetappprop MyServer -name adder -type ejb -resname adder/adder_eb -c C:¥home¥adder_ejb.xml
```

(3) 編集する属性設定項目

Enterprise Bean のセキュリティの定義（セキュリティアイデンティティ）の設定項目（<security-identity>）を次に示します。

項目	必須	対応するタグ名
説明	△	<description>
セキュリティアイデンティティ設定有無	△※	<use-caller-identity>
ロールのアイデンティティの説明	△	<run-as> - <description>
セキュリティロールの名称	△※	<run-as> - <role-name>
セキュリティロールに設定された名称	△	<run-as> - <user-name>

（凡例） △：任意

注※ セキュリティアイデンティティの設定をする場合，メソッド実行時に呼び出し元セキュリティアイデンティティを使用するかどうかで，次のように，どちらか一つ設定します。

- メソッド実行時に呼び出し元セキュリティアイデンティティを使用する場合
  セキュリティアイデンティティ設定有無（<use-caller-identity>）を設定します。
- メソッド実行時に呼び出し元セキュリティアイデンティティを使用しない場合
  ロールのアイデンティティ（<run-as>）情報を設定します。
- Message-driven Bean の場合は，ロールのアイデンティティ（<run-as>）情報だけ設定できます。

プロパティの設定項目については，次の個所を参照してください。

- マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.4.1 Session Bean 属性ファイルの指定内容」
- マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.5.1 Entity Bean 属性ファイルの指定内容」
- マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.6.1 MessageDrivenBean 属性ファイルの指定内容」

9.5.2　サーブレットと JSP のセキュリティアイデンティティ

サーブレットと JSP のセキュリティアイデンティティを定義します。

サーブレットが EJB 呼び出し時に使用する実行時アイデンティティ情報を設定します。

(1)　編集する属性ファイル

サーブレット属性ファイル

(2)　編集する属性ファイルの取得と属性の設定

- 属性ファイルの取得

  次に示すコマンドを実行してサーブレット属性ファイルを取得します。

  実行形式

```
cjgetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type war -resname <WAR表示名>/<サーブレットおよびJSPの表示名> -c <サーブレット属性ファイルパス>
```

  実行例

```
cjgetappprop MyServer -name adder -type war -resname adder/adder_sv -c C:¥home¥adder_war.xml
```

- 属性の設定

  次に示すコマンドを実行して，サーブレット属性ファイルの値を反映します。

  実行形式

```
cjsetappprop ［<サーバ名称>］ ［-nameserver <プロバイダURL>］ -name <J2EEアプリケーション名> -type war -resname <WAR表示名>/<サーブレットおよびJSPの表示名> -c <サーブレット属性ファイルパス>
```

  実行例

```
cjsetappprop MyServer -name adder -type war -resname adder/adder_sv -c C:¥home¥adder_war.xml
```

(3)　編集する属性設定項目

Web アプリケーション（サーブレットおよび JSP）のセキュリティの定義（セキュリティアイデンティティ）の設定項目を次に示します。

項目	必須	対応するタグ名
ロールのアイデンティティの説明	△	<run-as> - <description>
セキュリティロールの名称	○	<run-as> - <role-name>
セキュリティロールに設定された名称	○	<run-as> - <user-name>

（凡例）　○：必須　△：任意

プロパティの設定項目については，マニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」の「3.9.1　サーブレット属性ファイルの指定内容」を参照してください。

10 運用管理ポータルによる統合ユーザ管理の運用

この章では，Management Server の運用管理ポータルを利用した統合ユーザ管理の運用手順について説明します。

なお，運用管理ポータルの画面の操作方法，操作規則，操作時の注意事項などについては，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「7. Management Server の画面と基本操作」を参照してください。

10.1　この章の構成

統合ユーザ管理を設定すれば，統合ユーザ管理で使用しているリソースを監視したり，統合ユーザ管理機能で使用するリポジトリにあるユーザ情報を編集したりできます。この章では，Management Server の運用管理ポータルを利用した統合ユーザ管理での運用手順について説明します。

この章の構成を次の表に示します。

表 10-1　この章の構成（運用管理ポータルによる統合ユーザ管理の運用）

分類	タイトル	参照先
運用	統合ユーザ管理のリソースの監視	10.2
設定	リポジトリ管理によるユーザ情報の編集	10.3
注意事項	「リポジトリ管理」での注意事項	10.4

注　「解説」および「実装」について，この章での説明はありません。

10.2　統合ユーザ管理のリソースの監視

統合ユーザ管理を設定している場合は，統合ユーザ管理で使用しているリソースの状況や定義情報を確認できます。また，ログイン中のユーザのセッションを停止したり，接続プールの空き待ち監視数をリセットしたりする運用操作ができます。

10.2.1　統合ユーザ管理のリソース監視で確認できる項目

J2EE サーバで稼働している統合ユーザ管理のリソースについて，接続状況，稼働状況，および接続定義情報が確認できます。統合ユーザ管理のリソース監視で確認できる項目を次の表に示します。

表 10–2　統合ユーザ管理のリソース監視で確認できる項目

確認項目	説明
ログインセッションの監視	統合ユーザ管理のセッションの状態が参照できます。 • セッション ID • レルム名 • ユーザ ID • ログイン時刻 また，指定した統合ユーザ管理のセッションを停止することもできます。
LDAP 接続プールの監視	統合ユーザ管理が使用する，LDAP 接続プール状態が参照できます。ユーザ情報リポジトリとして LDAP ディレクトリサーバを使用している場合に参照できます。 • LDAP 設定番号 • 接続先 URL • 接続プールの定義数（最大値／最小値） • 接続プールの状態（接続数／利用数） • 接続プールの空き待ち監視（測定開始時刻／空き待ち最大数）
JDBC 接続プールの監視	統合ユーザ管理が使用する，JDBC 接続プール状態が参照できます。ユーザ情報リポジトリとしてデータベース（RDB）を使用している場合に参照できます。 • JDBC 設定番号 • 接続先 URL • 接続プールの定義数（最大値／最小値） • 接続プールの状態（接続数／利用数） • 接続プールの空き待ち監視（測定開始時刻／空き待ち最大数）
障害情報の表示	リソース監視で発生した障害について，その詳細情報が参照できます。

統合ユーザ管理のリソースを監視する手順を次に示します。

1. 運用管理ポータルにログインし，［運用管理ポータル］画面で「リソース監視」をクリックします。

2. 監視したい項目のモニタをクリックします。

   ログインセッションモニタ，LDAP 接続モニタ，または JDBC 接続モニタが選択できます。

10.2.2　操作する画面（統合ユーザ管理のリソース監視）

統合ユーザ管理のリソース監視で操作できる内容と操作する画面を次の表に示します。

表 10-3　統合ユーザ管理のリソース監視で操作できる内容と操作する画面

操作内容	操作	画面の参照先
ログインセッションモニタの表示	○	12.3.1
統合ユーザ管理のセッションの停止	○	12.3.2
LDAP 接続プールモニタの表示とリセット	○	12.4.1 12.4.2
LDAP 接続の定義情報の表示	○	12.4.3
LDAP 接続の障害情報の表示	○	12.4.4
JDBC 接続プールモニタの表示とリセット	○	12.5.1 12.5.2
JDBC 接続の定義情報の表示	○	12.5.3
JDBC 接続の障害情報の表示	○	12.5.4

（凡例）

○：任意。運用で必要と判断される場合に操作します。

10.3　リポジトリ管理によるユーザ情報の編集

統合ユーザ管理機能では，ユーザ情報を格納するリポジトリとして，ユーザ情報リポジトリ，およびシングルサインオン情報リポジトリを使用します。運用管理ポータルでは，ユーザ情報リポジトリ，およびシングルサインオン情報リポジトリ内のユーザ情報を編集できます。リポジトリごとにユーザ情報の編集手順を説明します。

10.3.1　ユーザ情報リポジトリに対するユーザ情報の編集

運用管理ポータルでは，ユーザ情報リポジトリとして使用する LDAP ディレクトリサーバに対して，次の操作ができます。

- LDAP ディレクトリサーバへの接続情報（バインド情報）の設定
- レルムの作成，および削除
- ユーザエントリのスキーマ定義
- ユーザエントリの作成，検索，削除，および編集

リポジトリ管理での操作は，LDAP ディレクトリサーバを使用していることが前提です。そのため，ユーザ管理を構築するために必要なスキーマ定義を含む LDAP ディレクトリサーバの設定は事前に行っておいてください。

なお，LDAP ディレクトリサーバとして Active Directory を使用して，ユーザの登録，ユーザ情報の更新，およびリポジトリ情報のパスワードの変更をする場合の方法については，「(2)　Active Directory を使用する場合」を参照してください。

(1)　Active Directory を使用しない場合

運用管理ポータルの「リポジトリ管理」を使用して，ユーザ情報リポジトリへユーザ情報を作成，検索，削除または編集するための手順を次に示します。

1. 運用管理ポータルにログインし，［運用管理ポータル］画面で「リポジトリ管理」をクリックします。

2. バインド情報を設定します。

   リポジトリ管理の［バインド情報の設定］画面で，LDAP ディレクトリサーバへ接続するための情報を設定します。

3. レルムを作成します。

   レルム管理の［レルムの作成］画面で，LDAP ディレクトリサーバにレルムを作成します。

4. ユーザエントリのスキーマを定義します。

   レルム管理の［ユーザエントリのスキーマ定義］画面で，レルムで管理するユーザエントリの属性と，ユーザのパスワードを保存するときの暗号化形式を設定します。

5. ユーザエントリを作成，検索，削除または編集します。

レルム管理の次の画面で，レルムにユーザエントリを作成，検索，削除または編集します。ユーザエントリの削除および編集は，ユーザエントリを検索してから実施します。

- ［ユーザエントリの作成］画面
- ［ユーザエントリの検索］画面
- ［ユーザエントリの削除］画面
- ［ユーザエントリの編集］画面

(2)　Active Directory を使用する場合

バインド情報の設定，ユーザエントリのスキーマ定義，ユーザエントリの作成については，「(1)　Active Directory を使用しない場合」の手順を参照してください。

SSL を使用して接続するための証明書の登録，および Active Directory 固有の環境設定の方法を次に示します。

(a)　証明書の登録

Management Server と Active Directory 間の通信で SSL を使用するための証明書を登録します。証明書の登録方法を次に示します。

1. 作成したデジタル証明書を，Active Directory がインストールされているサーバ（LDAP サーバ）に登録します。

   デジタル証明書の作成，登録方法については，Active Directory のドキュメントを参照してください。

2. Management Server に認証局（CA）の証明書を登録します。

   認証局の証明書の Management Server への登録は，Developer's Kit for Java に付属する keytool を使用して実行できます。keytool については，Java 2 SDK, Standard Edition のドキュメントを参照してください。keytool の実行例を次に示します。なお，表記の都合上，複数行にわたっていますが，実際は一行で記述します。

   Windows の場合

```
keytool -import -alias cakey -file C:¥temp¥cacer.cer -trustcacerts -keystore
”＜Application Serverのインストールディレクトリ＞¥jdk¥jre¥lib¥security¥cacerts”
```

   UNIX の場合

```
/opt/Cosminexus/jdk/bin/keytool -import -alias cakey -file /tmp/cacer.cer
-trustcacerts -keystore /opt/Cosminexus/jdk/jre/lib/security/cacerts
```

なお，keytool で証明書を登録する際に Management Server を起動していた場合は，Management Server を再起動してください。

(b) バインド情報の設定

リポジトリ管理の［バインド情報の設定］画面で必要な設定を次に示します。

- 「プロトコル」に「ldaps」を指定します。
- 「ポート」に「636」を指定します。
- ［Active Directory に接続する。］チェックボックスをチェックします。

(c) ユーザエントリのスキーマ定義

レルム管理の［ユーザエントリのスキーマ定義］画面で必要な設定を次に示します。

- デフォルトのオブジェクトクラス「inetorgperson」は削除して，オブジェクトクラスに「user」を追加します。
- ［必須属性の設定］の「ユーザ ID」に「cn」，「パスワード」に「unicodePwd」を指定します。
- ［任意属性の設定］で次の二つの属性を追加します。
  sAMAccountName
  userAccountControl

注意事項

- ［必須属性の設定］の「パスワード」に「userPassword」を指定した場合，SSL を使用しないでパスワードを変更できますが，Active Directory で管理するユーザのパスワードは変更できません。
- ［必須属性の設定］の「パスワード」に「unicodePwd」を指定した場合，「暗号化形式」の指定は無効になります。
- Active Directory では，既存ユーザのオブジェクトクラスの変更はサポートしていません。［バインド情報の設定］画面で［Active Directory に接続する。］チェックボックスにチェックして，［ユーザエントリのスキーマ定義］画面で新規にオブジェクトクラス「user」を追加したあとで，既存のユーザエントリを更新しても，ユーザエントリ作成時のオブジェクトクラスが適用されます。
- 「リポジトリ管理」の画面では，「ユーザ ID」に「sAMAccountName」を指定できません。

(d) ユーザエントリの作成

レルム管理の［ユーザエントリの作成］画面で設定します。

一つのユーザエントリを作成するためには，次に示す二つの設定が必要です。

- 「属性名」で「sAMAccountName」を選択して，「属性値」にセキュリティアカウントマネージャ（SAM）のアカウント名を指定します。なお，通常，「属性値」にはユーザ ID と同じ値を指定します。
- 「属性名」で「userAccountControl」を選択して，「属性値」にユーザアカウントのプロパティフラグを指定します。一般ユーザのユーザエントリを作成する場合は「512」を指定します。

注意事項

Active Directory がインストールされているサーバのセキュリティポリシーで，パスワードの長さが 1 文字以上に設定されている場合，「userAccountControl」の「属性値」に「512」を指定すると，ユーザを作成できません。この場合，次のどちらかの対処をして，ユーザエントリを作成してください。

- セキュリティポリシーのパスワードの長さを 0 文字以上に変更して，「userAccountControl」の「属性値」に「512」を指定する。
- セキュリティポリシーのパスワードの長さの設定は変更しないで，「userAccountControl」の「属性値」に「544」を指定する。

10.3.2 シングルサインオン情報リポジトリに対するシングルサインオン用のユーザ情報の編集

運用管理ポータルでは，シングルサインオン情報リポジトリとして使用する LDAP ディレクトリサーバに対して，次の操作ができます。

- シングルサインオン用の暗号鍵ファイルの設定
- シングルサインオン用のレルムの作成，および削除
- シングルサインオン用のユーザエントリのスキーマ定義
- シングルサインオン用のユーザエントリの作成，検索，削除，および編集

運用管理ポータルの「リポジトリ管理」を使用して，シングルサインオン情報リポジトリへシングルサインオン用のユーザ情報を作成，検索，削除または編集するための手順を次に示します。

1. **暗号鍵ファイルを適用します（任意）。**

   レルム管理の［暗号鍵ファイルの設定］画面で，シングルサインオン用のユーザ情報を暗号化，復号化するための暗号鍵ファイルを適用します。なお，暗号鍵ファイルはあらかじめ作成し，システム管理者が安全な方法で統合ユーザ管理機能を使用する J2EE サーバに配布しておいてください。

2. **シングルサインオン用のレルムを作成します。**

   レルム管理の［レルムの作成］画面で，LDAP ディレクトリサーバにレルムを作成します。

3. **シングルサインオン用のユーザエントリのスキーマを定義します。**

   レルム管理の［ユーザエントリのスキーマ定義（シングルサインオン用）］画面で，レルムで管理するユーザエントリの属性と，ユーザのパスワードを保存するときの暗号化形式を設定します。

4. **シングルサインオン用のユーザエントリを作成，検索，削除または編集します。**

   レルム管理の次の画面で，レルムにユーザエントリを作成，検索，削除または編集します。ユーザエントリの削除および編集は，ユーザエントリを検索してから実施します。

- ［ユーザエントリの作成（シングルサインオン用）］画面
- ［ユーザエントリの検索］画面
- ［ユーザエントリの削除］画面
- ［ユーザエントリの編集（シングルサインオン用）］画面

10.3.3　操作する画面（統合ユーザ管理のリポジトリ管理）

運用管理ポータルの「リポジトリ管理」で設定できる内容と操作する画面を次の表に示します。なお，シングルサインオンするかどうかによって，必要な操作が異なります。

表 10–4　統合ユーザ管理のリポジトリ管理で設定できる内容と操作する画面

設定内容	操作		画面の参照先
	シングルサインオンする	シングルサインオンしない	
バインド情報の設定	◎	◎	11.3.1
レルムの作成	◎	◎	11.4.1
レルムの削除	○	○	11.4.8
暗号鍵ファイルの設定	○	－	11.4.2
ユーザエントリのスキーマ定義	◎	◎	11.4.3
ユーザエントリの作成	◎	◎	11.4.5
ユーザエントリの検索	○	○	11.4.7
ユーザエントリの編集	○	○	11.4.9
ユーザエントリの削除	○	○	11.4.11
ユーザエントリのスキーマ定義（シングルサインオン用）	◎	－	11.4.4
ユーザエントリの作成（シングルサインオン用）	◎	－	11.4.6
ユーザエントリの編集（シングルサインオン用）	◎	－	11.4.10

（凡例）

◎：必要。最低限必要な操作です。

○：任意。運用で必要と判断される場合に操作します。

－：不要。必要な操作はありません。

10.4 「リポジトリ管理」での注意事項

リポジトリ管理での操作に関する注意事項を次に示します。

- リポジトリ管理での設定は一つの Web ブラウザで操作してください。複数の Web ブラウザで同時に操作しないでください。
- 各画面の値を直接入力できる入力フィールドでは，一部（パスワード/SecretData を入力するフィールド）を除いて指定した値の前後のスペースおよびタブは削除されます。

11 運用管理ポータルによるリポジトリ管理（統合ユーザ管理）

この章では，運用管理ポータルによるリポジトリ管理について説明します。リポジトリ管理は，統合ユーザ管理機能を使用する場合に必要な設定です。なお，注意事項については，「10.4 「リポジトリ管理」での注意事項」を参照してください。

なお，バッチアプリケーションを実行するシステムの場合，統合ユーザ管理機能は使用できません。

11.1　この章の構成

この章では，運用管理ポータルの「リポジトリ管理」に表示される画面の概要，操作手順および画面詳細について説明します。

この章の構成を次の表に示します。

表 11-1　この章の構成（運用管理ポータルによるリポジトリ管理）

分類	タイトル	参照先
解説	「リポジトリ管理」のツリーペインの構成	11.2
設定	リポジトリ管理	11.3
	レルム管理	11.4

注　「実装」,「運用」および「注意事項」について，この章での説明はありません。

11.2 「リポジトリ管理」のツリーペインの構成

この節では，「リポジトリ管理」のツリーペインの構成について説明します。

「リポジトリ管理」のツリーペインの構成を次に示します。

```
リポジトリ管理
│
└──レルム管理
    │
    ├──＜レルム名＞
    │
    └──＜レルム名（シングルサインオンだけ）＞
```

表 11-2 「リポジトリ管理」のツリーペインに表示されるノードの意味

ノード名	説明	ノード上でできる操作と参照先
リポジトリ管理	統合ユーザ管理のリポジトリ管理のルートです。ディレクトリサーバとの接続の設定ができます。	• バインド情報の設定（11.3.1 参照）
レルム管理	レルムのツリーです。この配下に，同一の認証ポリシーを適用する単位となるレルムが表示されます。 統合ユーザ管理のリポジトリにレルムを追加したり，シングルサインオン用の設定をしたりできます。	• レルムの作成（11.4.1 参照） • 暗号鍵ファイルの設定（11.4.2 参照）
＜レルム名＞	レルム名です。選択したレルムおよびそのレルムが管理しているユーザエントリに関する操作ができます。	• ユーザエントリのスキーマ定義（11.4.3 参照） • ユーザエントリの作成（11.4.5 参照） • ユーザエントリの検索（11.4.7 参照） • レルムの削除（11.4.8 参照） • ユーザエントリの編集（11.4.9 参照） • ユーザエントリの削除（11.4.11 参照）
＜レルム名（シングルサインオンだけ）＞	レルム名です。このレルムでは，ほかのシステムとシングルサインオンするために必要なシングルサインオン用情報リポジトリだけを管理しています。 選択したレルムおよびそのレルムが管理しているユーザエントリに関する操作ができます。	• ユーザエントリのスキーマ定義（シングルサインオン用）（11.4.4 参照） • ユーザエントリの作成（シングルサインオン用）（11.4.6 参照） • ユーザエントリの検索（11.4.7 参照） • レルムの削除（11.4.8 参照） • ユーザエントリの編集（シングルサインオン用）（11.4.10 参照） • ユーザエントリの削除（11.4.11 参照）

11.3　リポジトリ管理

この節では，LDAP ディレクトリサーバと接続するための設定で使用する次の画面の機能概要，表示手順，操作手順，画面詳細について説明します。

- ［バインド情報の設定］画面

11.3.1　バインド情報の設定

［バインド情報の設定］画面を次の図に示します。

図 11–1　［バインド情報の設定］画面

(1)　機能概要

LDAP ディレクトリサーバと接続（バインド）するための情報を設定します。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［リポジトリ管理］をクリックします。

3. ［バインド情報の設定］タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. ［ホスト名］，［ポート］などを指定します。

2. ［適用］ボタンをクリックします。
   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

プロトコル（必須）

LDAP ディレクトリサーバに接続するときのプロトコルを指定します。

ホスト名（必須）

接続する LDAP ディレクトリサーバのホスト名を指定します。

ポート（必須）

接続する LDAP ディレクトリサーバのポート番号を指定します。

バインド DN（必須）

接続する LDAP ディレクトリサーバのバインド DN を指定します。

パスワード（必須）

接続する LDAP ディレクトリサーバのパスワードを指定します。画面では「*」で表示されます。

ベース DN（必須）

レルムを管理するベース DN を指定します。

Active Directory に接続する。

LDAP ディレクトリサーバとして Active Directory を使用する場合は，このチェックボックスをチェックしてください。Active Directory 以外を使用する場合，チェックは不要です。

［適用］ボタン

指定した内容を設定します。

11.4　レルム管理

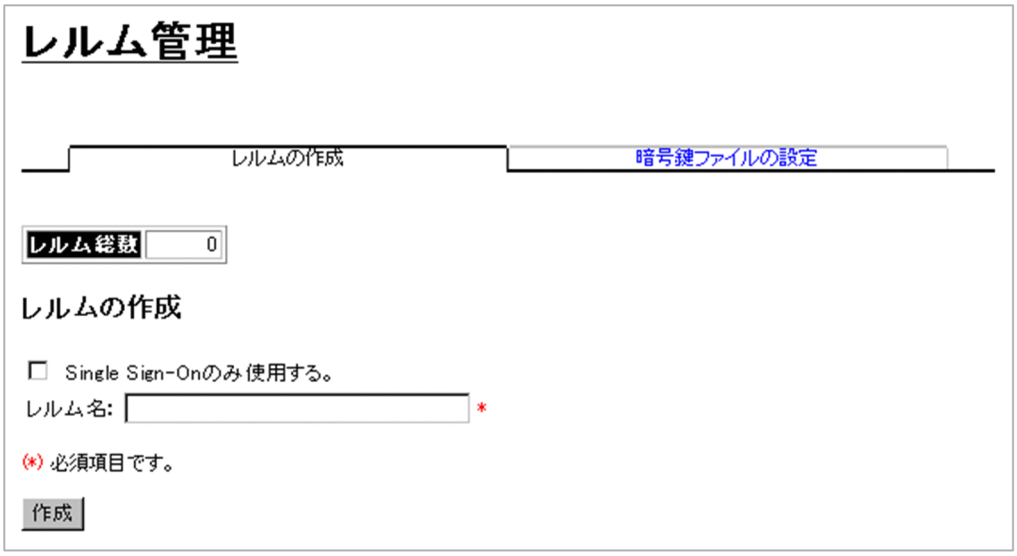
この節では，レルムの作成，ユーザエントリのスキーマ定義，ユーザエントリの作成，編集，削除などで使用する次の画面の機能概要，表示手順，操作手順，画面詳細について説明します。

- ［レルムの作成］画面
- ［暗号鍵ファイルの設定］画面
- ［ユーザエントリのスキーマ定義］画面
- ［ユーザエントリのスキーマ定義（シングルサインオン用)］画面
- ［ユーザエントリの作成］画面
- ［ユーザエントリの作成（シングルサインオン用)］画面
- ［ユーザエントリの検索］画面
- ［レルムの削除］画面
- ［ユーザエントリの編集］画面
- ［ユーザエントリの編集（シングルサインオン用)］画面
- ［ユーザエントリの削除］画面

11.4.1　レルムの作成

［レルムの作成］画面を次の図に示します。

図 11-2　［レルムの作成］画面



(1) 機能概要

LDAP ディレクトリサーバにレルムを作成します。レルムとは，同一の認証ポリシーを適用する範囲のことです。

作成に成功すると，ツリーペインの［レルム管理］下にレルムが追加されます。追加されたレルムを確認するには，ツリーペインの［ツリーの初期化］アンカーをクリックするか，または［レルム管理］アイコンをダブルクリックします。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］をクリックします。
3. ［レルムの作成］タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. ［レルム名］などを指定します。
2. ［作成］ボタンをクリックします。
   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

レルム総数

統合ユーザ管理が管理するユーザ情報リポジトリのレルムとシングルサインオン情報リポジトリのレルムを合わせた数が表示されます。

Single Sign-On のみ使用する。

チェックすると，シングルサインオン情報リポジトリだけが操作できるレルムを作成します。

注意事項

［Single Sign-On のみ使用する。］をチェックしてレルムを作成すると，作成したレルムではシングルサインオンを使用するための情報だけが管理できます。このレルムは，統合ユーザ管理以外でユーザ管理をしているアプリケーションと接続するためのユーザ情報を定義する場合に使用します。

［Single Sign-On のみ使用する。］をチェックして作成したレルムでは，あとから統合ユーザ管理のユーザ管理をしようとしても追加できません。該当レルムの再作成が必要になります。そのため，

将来統合ユーザ管理でユーザ管理をする予定がある場合は，［Single Sign-On のみ使用する。］のチェックを外して作成することを推奨します。

レルム名（必須）

ユーザ情報リポジトリまたはシングルサインオン情報リポジトリに作成するレルム名を指定します。指定は必須です。

レルム名は，英数字で指定します。大文字と小文字は区別されません。DN 名で使用できる名前を付けてください。また，「mappings」は予約語のため指定しないでください。

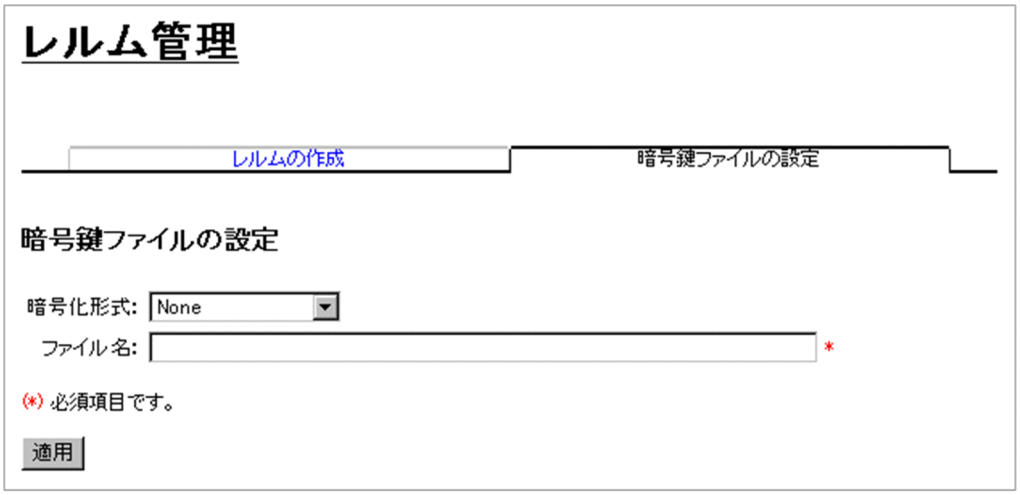
［作成］ボタン

指定した内容でレルムを作成します。

11.4.2　暗号鍵ファイルの設定

［暗号鍵ファイルの設定］画面を次の図に示します。

図 11–3　［暗号鍵ファイルの設定］画面



(1)　機能概要

シングルサインオン用認証情報の暗号化データ（SecretData）を暗号化するための暗号鍵ファイルを指定します。暗号鍵ファイルの設定は，シングルサインオン用認証情報を定義する場合だけ行います。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］をクリックします。
3. ［暗号鍵ファイルの設定］タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. ［暗号化形式］，［ファイル名］などを指定します。

2. ［適用］ボタンをクリックします。

   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

暗号化形式

シングルサインオン用認証情報を暗号化する方法を指定します。

- JCE：JCE を使用して暗号化します。
- None：暗号化しません。

ファイル名（必須）

作成または適用する暗号鍵ファイルのファイル名を絶対パスで指定します。

（指定例）

- Windows の場合
  C:¥APS¥config¥DESKeyfile.key
- UNIX の場合
  /opt/APS/config/DESKeyfile.key

指定したファイルがない場合は作成し，ある場合はそのファイルを適用します。なお UNIX の場合，作成される暗号鍵ファイルのパーミッションは Management Server 起動時の umask の設定に依存します。このため，一般ユーザに暗号鍵ファイルを参照されないように，作成後に適切なパーミッションに変更するか，または適切なパーミッションが設定されたディレクトリに作成してください。

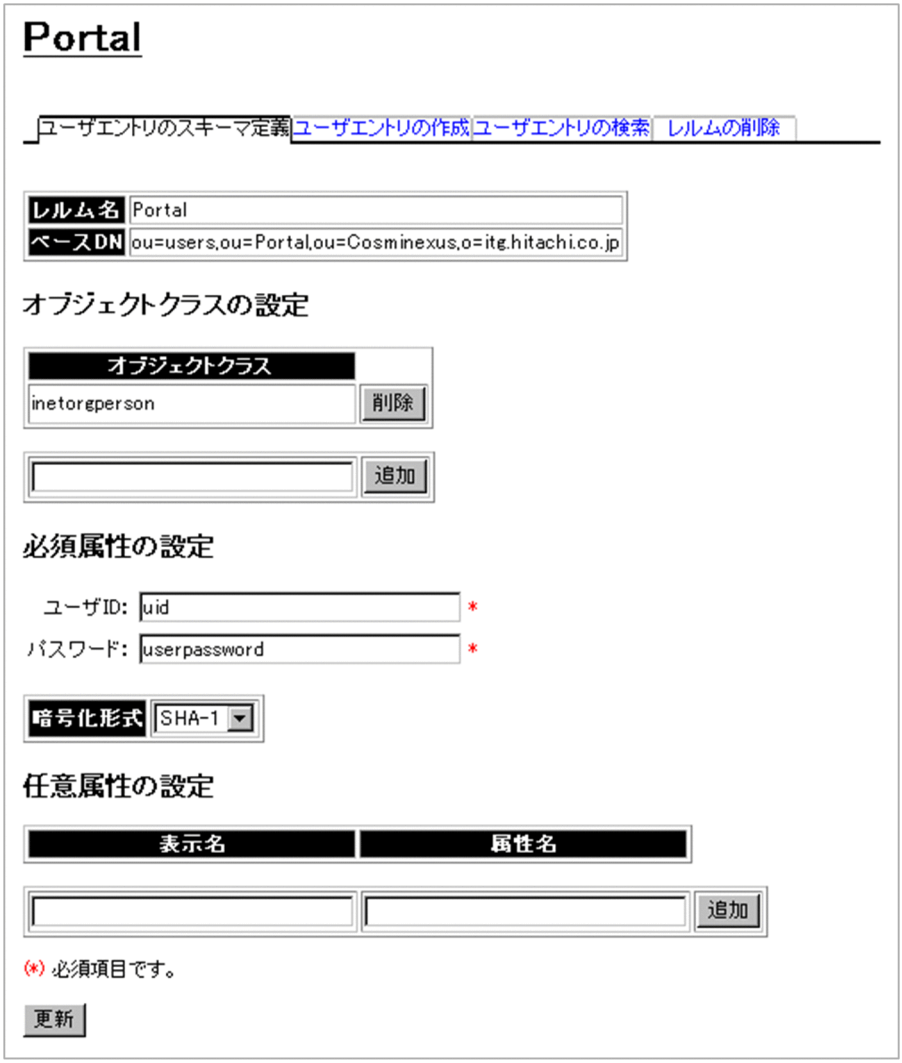
［適用］ボタン

指定した内容で暗号鍵ファイルを作成または適用します。

11.4.3 ユーザエントリのスキーマ定義

［ユーザエントリのスキーマ定義］画面を次の図に示します。

図 11-4　［ユーザエントリのスキーマ定義］画面



(1)　機能概要

レルムで管理するユーザエントリの属性と，ユーザのパスワードを保存するときの暗号化形式を設定します。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。
3. ［ユーザエントリのスキーマ定義］タブをクリックします。

(3)　操作手順

画面での操作手順を次に示します。

1. ［ユーザ ID］，［パスワード］などを指定します。

   オブジェクトクラスや任意属性を追加する場合，［オブジェクトクラスの設定］や［任意属性の設定］に項目を指定したあとに［追加］ボタンをクリックします。

2. ［更新］ボタンをクリックします。

   ［オブジェクトクラスの設定］や［任意属性の設定］で追加または削除した情報は，［更新］ボタンをクリックしないと有効になりません。必ず［更新］ボタンをクリックしてください。

   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

レルム名

レルム名が表示されます。

ベース DN

レルムのベース DN が表示されます。

オブジェクトクラスの設定

ユーザエントリに追加する属性のオブジェクトクラスの一覧が表示されます。デフォルトは「inetorgperson」です。「inetorgperson」には，属性「cn」と「sn」が必須属性として割り当てられています。必須属性または任意属性の属性名に「cn」または「sn」を指定しない場合は，ユーザ ID が仮定されます。

オブジェクトクラスを追加する場合は，入力フィールドにオブジェクトクラス名を指定し，［追加］ボタンをクリックします。また，追加したオブジェクトクラスに必須属性として割り当てられている属性名を，必須属性または任意属性の属性名に指定してください。

オブジェクトクラスを削除する場合は，削除するオブジェクトクラスの［削除］ボタンをクリックします。

［追加］ボタン

［オブジェクトクラスの設定］で指定した内容を有効にします。

［削除］ボタン

［オブジェクトクラスの設定］に指定されている内容を削除します。

必須属性の設定

ユーザエントリに必ず追加する属性を指定します。

ユーザ ID（必須）

ユーザ ID の属性名を指定します。デフォルトは「uid」です。パスワードおよび任意属性と同一の属性名は指定できません。

パスワード（必須）

パスワードの属性名を指定します。デフォルトは「userpassword」です。ユーザ ID および任意属性と同一の属性名は指定できません。

暗号化形式

［必須属性の設定］のパスワードに設定した属性の暗号化の形式を指定します。

- SHA-1：SHA-1 形式で暗号化します。
- SHA-224：SHA-224 形式で暗号化します。
- SHA-256：SHA-256 形式で暗号化します。
- SHA-384：SHA-384 形式で暗号化します。
- SHA-512：SHA-512 形式で暗号化します。
- MD5：MD5 形式で暗号化します。
- None：暗号化しません。

デフォルトは「SHA-1」です。

任意属性の設定

ユーザエントリに属性を追加する場合，追加する属性名と表示名を指定して［追加］ボタンをクリックします。属性名を指定して表示名を省略した場合，表示名には属性名が仮定されます。ユーザ ID およびパスワードと同一の属性名は指定できません。指定した属性を削除する場合は，削除する属性の［削除］ボタンをクリックします。

ここで設定した属性には，ユーザエントリの追加時に属性値を指定できます。ただし，リポジトリ管理ではサブコンテキストの属性は作成できません。また，バイナリを扱う属性も作成できません。

［更新］ボタン

指定した内容で定義します。

11.4.4　ユーザエントリのスキーマ定義（シングルサインオン用）

［ユーザエントリのスキーマ定義（シングルサインオン用）］画面を次の図に示します。

図 11-5 ［ユーザエントリのスキーマ定義（シングルサインオン用）］画面

(1) 機能概要

ユーザエントリを追加または削除，および暗号化データ（SecretData）を変更した場合に，他システムと同期を取るためのリスナクラスを設定します。

注意事項

リポジトリ管理では，イベントリスナクラスのインスタンスを取得する際，デフォルトコンストラクタを呼び出します。そのため，リポジトリ管理で使用するイベントリスナクラスでは，オーバーライドされたコンストラクタで初期化処理などをしないでください。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。
   このとき，レルム作成時に［Single Sign-On のみ使用する。］をチェックしたレルム名をクリックしてください。
3. ［ユーザエントリのスキーマ定義］タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. ［リスナクラス名］を指定して，［追加］ボタンをクリックします。
2. ［更新］ボタンをクリックします。

［リスナクラスの登録］で追加または削除した情報は，［更新］ボタンをクリックしないと有効になりません。必ず［更新］ボタンをクリックしてください。

エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

レルム名

レルム名が表示されます。

ベース DN

レルムのベース DN が表示されます。

リスナクラスの登録

イベントリスナクラスのクラス名を完全限定名で指定し，［追加］ボタンをクリックします。イベントリスナクラスは複数登録できます。削除する場合は，削除するリスナクラスの［削除］ボタンをクリックします。

指定するイベントリスナクラスは，統合ユーザ管理フレームワークが提供する SSODataListener インタフェースを実装し，次のディレクトリに格納しておいてください。

- Windows の場合
  ＜ Application Server のインストールディレクトリ＞
  ¥manager¥containers¥m¥j2eeapps¥mngsvr¥WEB-INF¥classes
- UNIX の場合
  /opt/Cosminexus/manager/containers/m/j2eeapps/mngsvr/WEB-INF/classes

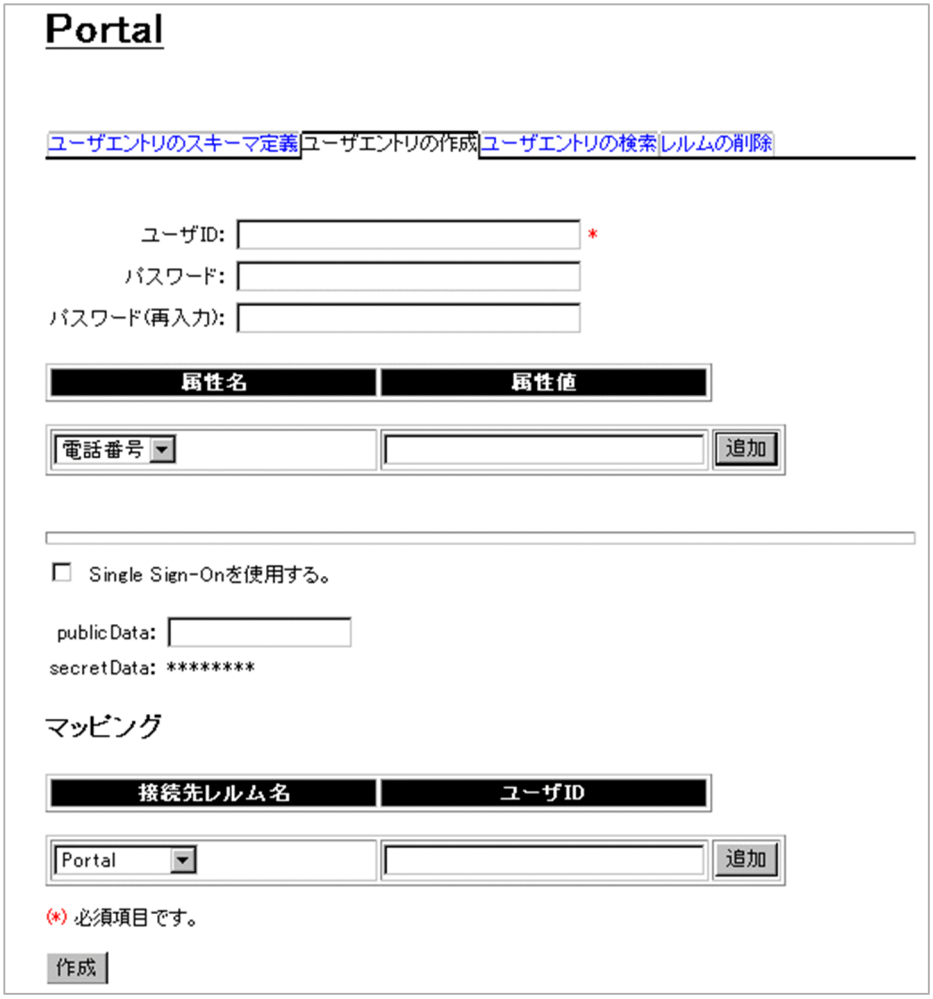
［更新］ボタン

指定した内容で定義します。

11.4.5 ユーザエントリの作成

［ユーザエントリの作成］画面を次の図に示します。

図 11-6 ［ユーザエントリの作成］画面



(1) 機能概要

ツリーペインで選択されているレルムにユーザエントリを作成します。指定できる文字列および長さについては，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「7.5.3 「リポジトリ管理」での規則」を参照してください。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。
3. ［ユーザエントリの作成］タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. ［ユーザ ID］，［パスワード］などを指定します。

   属性値やユーザ ID を指定する場合，項目を指定したあとに［追加］ボタンをクリックします。

2. ［作成］ボタンをクリックします。

   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

ユーザ ID（必須）

ユーザ ID を指定します。ユーザ ID には，各レルムで一意な文字列を指定してください。指定は必須です。

パスワード

パスワードを指定します。画面では「*」で表示されます。

パスワード（再入力）

指定したパスワードを確認するために，パスワードを再度指定します。

属性名

「11.4.3　ユーザエントリのスキーマ定義」の［任意属性の設定］で定義した属性の表示名が表示されます。

属性値

属性名に対応する属性値を指定します。

属性値を追加する場合は，メニューから属性名を選択し，属性値を指定して［追加］ボタンをクリックします。一つの属性名に複数の属性値が設定できます。属性値を削除する場合は，削除する属性値の［削除］ボタンをクリックします。属性値を変更する場合は，該当属性を一度削除し，変更後の属性を追加してください。

［追加］ボタン

［属性名］および［属性値］で指定した内容を有効にします。

［削除］ボタン

［属性名］および［属性値］に指定されている内容を削除します。

Single Sign-On を使用する。

このチェックボックスをチェックした場合，該当ユーザのシングルサインオンの情報が更新されます。チェックしない場合は，シングルサインオンの情報が指定されていても更新されません。

publicData

シングルサインオン用認証情報の非暗号化データを指定します。指定は任意です。

secretData

シングルサインオン用認証情報の暗号化データが「*」で表示されます。このフィールドは，自動的にパスワードと同期するため，指定はできません。

マッピング

シングルサインオンを使用する場合に，接続先のレルム名とユーザ ID を指定します。選択できるレルムは，シングルサインオン情報リポジトリにある，統合ユーザ管理フレームワークが推奨する DIT 構成に準拠したレルムです。指定は任意です。指定する場合は，接続先レルム名とユーザ ID を指定して，［追加］ボタンをクリックします。削除する場合は，削除するユーザ ID の［削除］ボタンをクリックします。

接続先レルム名

接続先のレルム名をメニューから選択します。

ユーザ ID

接続先のレルムのユーザ ID を指定します。

［追加］ボタン

［接続先レルム名］および［ユーザ ID］で指定した内容を有効にします。

［削除］ボタン

［接続先レルム名］および［ユーザ ID］に指定されている内容を削除します。

［作成］ボタン

指定した内容でユーザエントリを作成します。

11.4.6　ユーザエントリの作成（シングルサインオン用）

［ユーザエントリの作成（シングルサインオン用）］画面を次の図に示します。

図 11-7　［ユーザエントリの作成（シングルサインオン用）］画面

(1) 機能概要

シングルサインオン情報リポジトリにユーザエントリを作成します。指定できる文字列および長さについては，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「7.5.3 「リポジトリ管理」での規則」を参照してください。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。
   このとき，レルム作成時に［Single Sign-On のみ使用する。］をチェックしたレルム名をクリックしてください。
3. ［ユーザエントリの作成］タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. ［ユーザ ID］，［publicData］などを指定します。
   マッピングを指定する場合，項目を指定したあとに［追加］ボタンをクリックします。
2. ［作成］ボタンをクリックします。
   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

ユーザ ID（必須）

ユーザ ID を指定します。ユーザ ID には，各レルムで一意な文字列を指定してください。指定は必須です。

publicData

シングルサインオン用認証情報の非暗号化データを指定します。指定は任意です。

secretData

シングルサインオン用認証情報の暗号化データを指定します。画面では「*」で表示されます。

secretData（再入力）

secretData に指定した内容を再度指定します。

マッピング

接続先のレルム名とユーザ ID を指定します。選択できるレルムは，シングルサインオン情報リポジトリにある，統合ユーザ管理フレームワークが推奨する DIT 構成に準拠したレルムです。指定は任意です。指定する場合は，接続先レルム名とユーザ ID を指定して，［追加］ボタンをクリックします。削除する場合は，削除する情報の［削除］ボタンをクリックします。

接続先レルム名

接続先のレルム名をメニューから選択します。

ユーザ ID

接続先のレルムのユーザ ID を指定します。

［追加］ボタン

［接続先レルム名］および［ユーザ ID］で指定した内容を有効にします。

［削除］ボタン

［接続先レルム名］および［ユーザ ID］に指定されている内容を削除します。

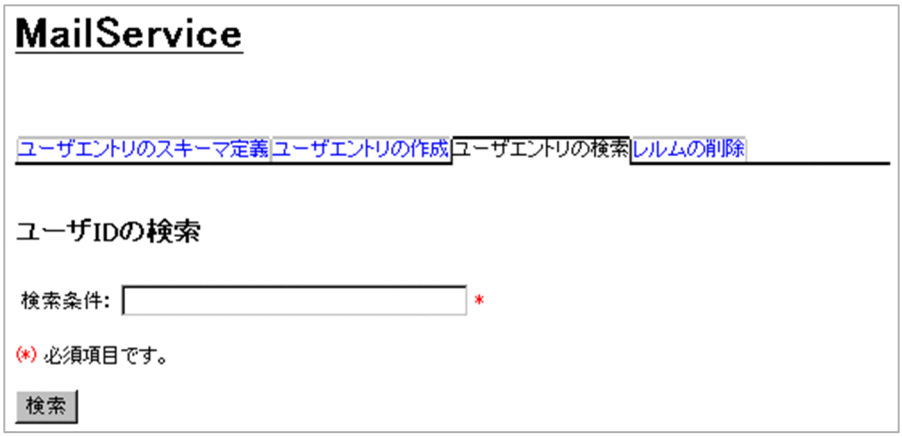
［作成］ボタン

指定した内容でユーザエントリを作成します。

11.4.7　ユーザエントリの検索

［ユーザエントリの検索］画面を次の図に示します。

図 11-8　［ユーザエントリの検索］画面



(1)　機能概要

ユーザエントリを検索します。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。

2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。

3. ［ユーザエントリの検索］タブをクリックします。

(3)　操作手順

画面での操作手順を次に示します。

1. ［検索条件］を指定します。

2. ［検索］ボタンをクリックします。

   検索結果が［ユーザエントリの検索］画面の下部に表示されます。検索結果にはリンクが設定され，クリックすると［ユーザエントリの編集］画面が表示されます。エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

   検索結果が表示された［ユーザエントリの検索］画面を次の図に示します。

図 11-9　検索結果が表示された［ユーザエントリの検索］画面

(4)　画面詳細

画面に表示される項目およびボタンについて説明します。

検索条件（必須）

検索するユーザ ID を指定します。検索対象は，ツリーペインで選択されているレルムが管理するユーザエントリのユーザ ID です。検索条件には「*」（ワイルドカード）が使用できます。

（指定例）taro*：「taro」で始まるユーザ ID を検索します。

検索条件によっては，対象となるユーザエントリ数が多くなり表示に時間が掛かる場合があります。

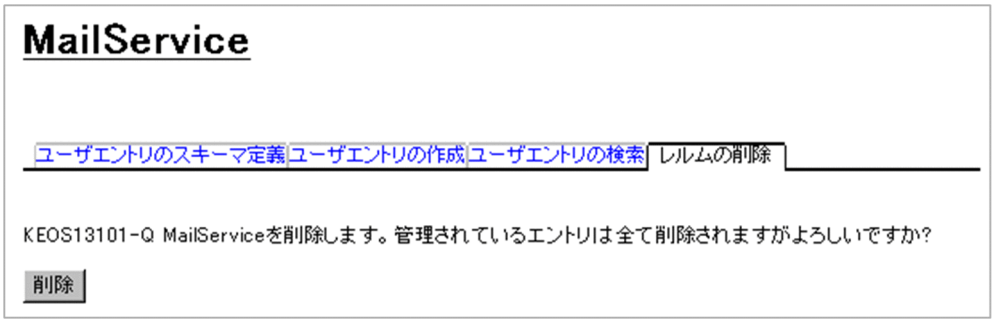
［検索］ボタン

指定した内容で検索します。

11.4.8　レルムの削除

［レルムの削除］画面を次の図に示します。

図 11-10　［レルムの削除］画面



(1)　機能概要

LDAP ディレクトリサーバのレルムを削除します。該当レルムが管理するユーザエントリはすべて削除されます。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。
3. ［レルムの削除］タブをクリックします。

(3)　操作手順

画面での操作手順を次に示します。

1. 表示内容を確認して，［削除］ボタンをクリックします。

   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4)　画面詳細

画面に表示される項目およびボタンについて説明します。

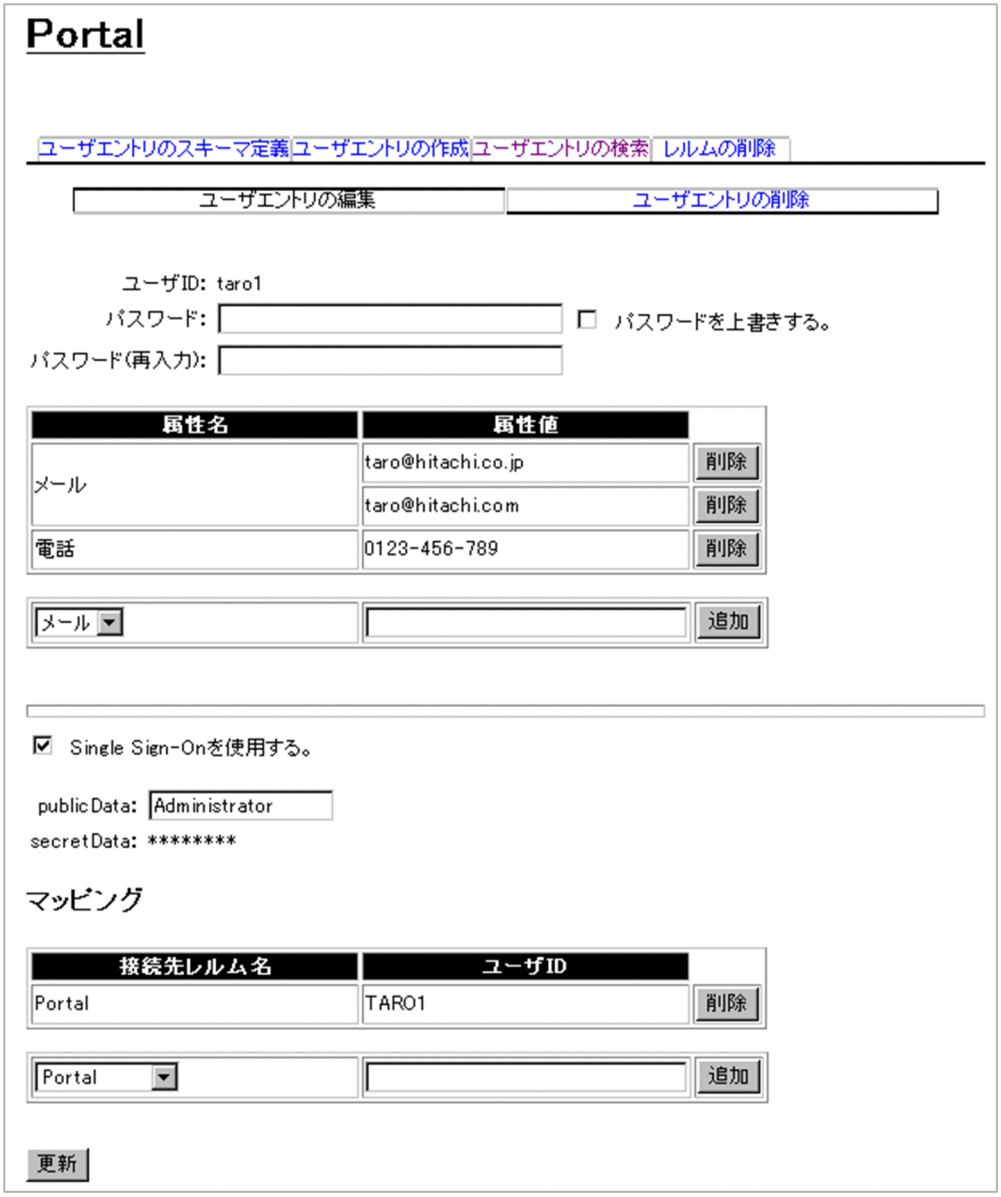
［削除］ボタン

該当レルムが管理するすべてのユーザエントリ，および LDAP ディレクトリサーバのレルムを削除します。

11.4.9　ユーザエントリの編集

［ユーザエントリの編集］画面を次の図に示します。

図 11-11　［ユーザエントリの編集］画面



(1)　機能概要

ユーザエントリを編集します。指定できる文字列および長さについては，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「7.5.3 「リポジトリ管理」での規則」を参照してください。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。
3. ［ユーザエントリの検索］タブをクリックします。
4. 編集するユーザ ID を［検索条件］に指定して，［検索］ボタンをクリックします。
5. 検索結果として表示されたユーザ ID をクリックします。
6. ［ユーザエントリの編集］タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. ［パスワード］などを指定します。
   設定内容を追加する場合，項目を指定したあとに［追加］ボタンをクリックします。設定内容を削除する場合，［削除］ボタンをクリックします。
2. ［更新］ボタンをクリックします。
   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

ユーザ ID

ユーザ ID は変更できません。変更したい場合は，ユーザエントリを LDAP ディレクトリサーバから削除し，再度作成してください。

パスワード

パスワードを変更する場合，変更後のパスワードを指定します。

パスワード（再入力）

指定したパスワードを確認するために，パスワードを再度指定します。

パスワードを上書きする

指定したパスワードで既存のエントリのパスワードを上書きする場合は，このチェックボックスをチェックしてください。チェックしない状態では，指定したパスワードは有効になりません。

属性名

属性名が表示されます。「11.4.3　ユーザエントリのスキーマ定義」の［任意属性の設定］で定義していない属性の場合，［属性名］には LDAP ディレクトリサーバの属性名（cn など）が表示されます。

属性値

属性名に対応する属性値が表示されます。一つの属性名に複数の属性値が設定できます。

属性値を追加するには，メニューから属性名を選択し，属性値を指定して［追加］ボタンをクリックします。属性値を削除するには，削除する属性値の［削除］ボタンをクリックします。属性値を変更する場合は，該当属性を一度削除し，変更後の属性を追加してください。

［追加］ボタン

［属性名］および［属性値］で指定した内容を有効にします。

［削除］ボタン

［属性名］および［属性値］に指定されている内容を削除します。

Single Sign-On を使用する。

シングルサインオン情報リポジトリにユーザエントリがある場合は，このチェックボックスは自動的にチェックされます。シングルサインオンの情報を変更する場合は，チェックした状態で変更してください。

publicData

シングルサインオン用認証情報の非暗号化データを変更する場合，変更後のデータを指定します。

secretData

シングルサインオン用認証情報の暗号化データが「*」で表示されます。

このフィールドは，自動的にパスワードと同期するため，指定はできません。

マッピング

接続先のレルム名とユーザ ID を変更する場合，変更後の情報を指定します。追加する場合は，接続先レルム名とユーザ ID を指定して［追加］ボタンをクリックします。削除する場合は，削除するユーザ ID の［削除］ボタンをクリックします。

接続先レルム名

接続先のレルム名をメニューから選択します。

ユーザ ID

接続先のレルムのユーザ ID を指定します。指定は任意です。

［更新］ボタン

指定した内容でユーザエントリを更新します。

11.4.10　ユーザエントリの編集（シングルサインオン用）

［ユーザエントリの編集（シングルサインオン用）］画面を次の図に示します。

図 11-12 ［ユーザエントリの編集（シングルサインオン用）］画面

(1) 機能概要

ユーザエントリを編集します。指定できる文字列および長さについては，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「7.5.3 「リポジトリ管理」での規則」を参照してください。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。
   このとき，レルム作成時に［Single Sign-On のみ使用する。］をチェックしたレルム名をクリックしてください。
3. ［ユーザエントリの検索］タブをクリックします。
4. 編集するユーザ ID を［検索条件］に指定して，［検索］ボタンをクリックします。
5. 検索結果として表示されたユーザ ID をクリックします。
6. ［ユーザエントリの編集］タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. [publicData] などを指定します。

   設定内容を追加する場合，項目を指定したあとに［追加］ボタンをクリックします。設定内容を削除する場合，［削除］ボタンをクリックします。

2. [更新] ボタンをクリックします。

   エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

ユーザ ID

ユーザ ID は変更できません。変更したい場合は，ユーザエントリを LDAP ディレクトリサーバから削除し，再度作成してください。

publicData

シングルサインオン用認証情報の非暗号化データを変更する場合，変更後のデータを指定します。

secretData

シングルサインオン用認証情報の暗号化データを変更する場合，変更後のデータを指定します。画面では「*」で表示されます。

secretData（再入力）

secretData を変更した場合，指定した内容を再度指定します。

secretData を上書きする

指定した secretData で既存のエントリのパスワードを上書きする場合は，このチェックボックスをチェックしてください。チェックしない状態では，指定した secretData は有効になりません。

マッピング

接続先のレルム名とユーザ ID が表示されます。変更する場合は，変更後の情報を指定します。選択できるレルムは，シングルサインオン情報リポジトリにある，統合ユーザ管理フレームワークが推奨する DIT 構成に準拠したレルムです。追加する場合は，接続先レルム名とユーザ ID を指定して，［追加］ボタンをクリックします。削除する場合は，削除する情報の［削除］ボタンをクリックします。

接続先レルム名

接続先のレルム名をメニューから選択します。

ユーザ ID

接続先のレルムのユーザ ID を指定します。指定は任意です。

[追加] ボタン

［接続先レルム名］および［ユーザ ID］で指定した内容を有効にします。

［削除］ボタン

［接続先レルム名］および［ユーザ ID］に指定されている内容を削除します。

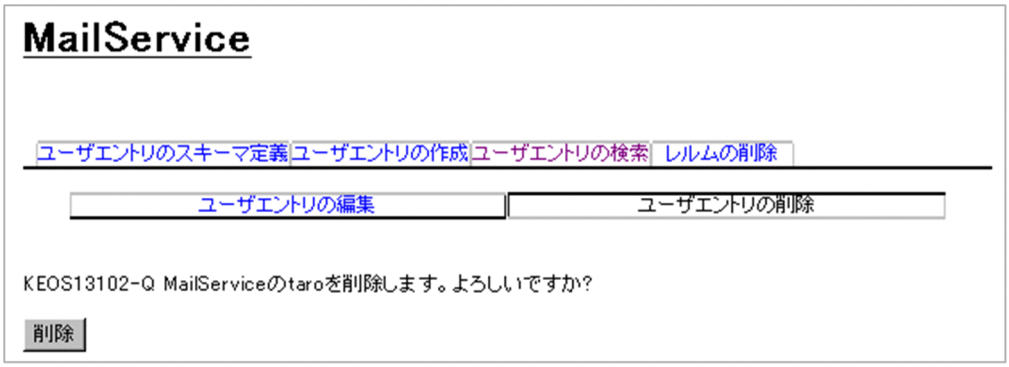
［更新］ボタン

指定した内容でユーザエントリを更新します。

11.4.11　ユーザエントリの削除

［ユーザエントリの削除］画面を次の図に示します。

図 11-13　［ユーザエントリの削除］画面



(1)　機能概要

ユーザエントリを削除します。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで［リポジトリ管理］アンカーをクリックします。
2. ツリーペインで［レルム管理］－［<レルム名>］をクリックします。
3. ［ユーザエントリの検索］タブをクリックします。
4. 削除するユーザ ID を［検索条件］に指定して，［検索］ボタンをクリックします。
5. 検索結果として表示されたユーザ ID をクリックします。
6. ［ユーザエントリの削除］タブをクリックします。

(3)　操作手順

画面での操作手順を次に示します。

1. 表示内容を確認して，［削除］ボタンをクリックします。

エラーが表示された場合は，［戻る］アンカーをクリックして原因を取り除き，再度実行します。

(4) 画面詳細

画面に表示される項目およびボタンについて説明します。

［削除］ボタン

ユーザエントリを削除します。

12 リソース監視（統合ユーザ管理）

この章では，運用管理ポータルの「リソース監視」に関する操作方法について説明します。［リソース監視］画面は，稼働中の J2EE サーバで実行されている統合ユーザ管理のリソースを監視するための画面です。

なお，バッチアプリケーションを実行するシステムの場合，統合ユーザ管理機能は使用できません。

12.1　この章の構成

この章では，運用管理ポータルの「リソース監視」に関する操作方法について説明します。

この章の構成を次の表に示します。

表 12–1　この章の構成（リソース監視（統合ユーザ管理））

分類	タイトル	参照先
解説	「リソース監視」のツリーペインの構成	12.2
設定	ログインセッションの監視	12.3
	LDAP 接続モニタの監視	12.4
	JDBC 接続モニタの監視	12.5
注意事項	LDAP 接続の障害情報の表示	12.4.4
	JDBC 接続の障害情報の表示	12.5.4

注　「実装」および「運用」について，この章での説明はありません。

12.2 「リソース監視」のツリーペインの構成

この節では，「リソース監視」のツリーペインの構成について説明します。

ツリーペインは，次に示すビューで構成されています。

- ホストビュー
- サーバビュー

12.2.1 「リソース監視」のホストビューの構成

「リソース監視」のホストビューには，操作できる画面がありません。「リソース監視」は，サーバビューで操作してください。

12.2.2 「リソース監視」のサーバビューの構成

「リソース監視」のサーバビューの構成は，次のとおりです。

```
リソース監視
 │
 └──<運用管理ドメイン名>
     └──論理J2EEサーバ
         ├──J2EEサーバ
         │   └──<J2EEサーバ名>
         │       ├──ログインセッションモニタ
         │       ├──LDAP接続モニタ
         │       └──JDBC接続モニタ
         │
         └──J2EEサーバクラスタ
             └──<J2EEサーバクラスタ名>
                 └──<J2EEサーバ名>
                     ├──ログインセッションモニタ
                     ├──LDAP接続モニタ
                     └──JDBC接続モニタ
```

サーバビューに表示されるノードの意味を次に示します。

表 12-2 「リソース監視」のサーバビューに表示されるノードの意味

ノード名	説明	ノード上でできる操作と参照先
<運用管理ドメイン名>	運用管理ドメインに定義されている論理サーバのルートです。運用管理ドメイン内のすべての論理サーバ（J2EE サーバ）を対象にした操作ができます。	• 運用管理ドメイン全体の論理サーバのステータス監視（マニュアル「アプリケーションサーバ運用管理ポータル操作ガイド」の 13.2.2 参照）

ノード名	説明	ノード上でできる操作と参照先
論理 J2EE サーバ	J2EE サーバおよび J2EE サーバクラスタのルートです。	このノードには操作画面がありません。
J2EE サーバ	J2EE サーバのツリーです。	このノードには操作画面がありません。
＜ J2EE サーバ名＞	各 J2EE サーバ名です。	このノードには操作画面がありません。
ログインセッションモニタ	選択した J2EE サーバのログインセッションモニタを操作できます。	• ログインセッションモニタの表示（12.3.1 参照） • 統合ユーザ管理のセッションの停止（12.3.2 参照）
LDAP 接続モニタ	選択した J2EE サーバの LDAP 接続モニタを操作できます。	• LDAP 接続プールモニタの表示（12.4.1 参照） • LDAP 接続プールの空き待ち監視のリセット（12.4.2 参照） • LDAP 接続の定義情報の表示（12.4.3 参照） • LDAP 接続の障害情報の表示（12.4.4 参照）
JDBC 接続モニタ	選択した J2EE サーバの JDBC 接続モニタを操作できます。	• JDBC 接続プールモニタの表示（12.5.1 参照） • JDBC 接続プールの空き待ち監視のリセット（12.5.2 参照） • JDBC 接続の定義情報の表示（12.5.3 参照） • JDBC 接続の障害情報の表示（12.5.4 参照）
J2EE サーバクラスタ	J2EE サーバクラスタのツリーです。	このノードには操作画面がありません。
＜ J2EE サーバクラスタ名＞	各 J2EE サーバクラスタ名です。	このノードには操作画面がありません。
＜ J2EE サーバ名＞	J2EE サーバクラスタ内の各 J2EE サーバ名です。	このノードには操作画面がありません。
ログインセッションモニタ	J2EE サーバクラスタ内の選択した J2EE サーバのログインセッションモニタを操作できます。	• ログインセッションモニタの表示（12.3.1 参照） • 統合ユーザ管理のセッションの停止（12.3.2 参照）
LDAP 接続モニタ	J2EE サーバクラスタ内の選択した J2EE サーバの LDAP 接続モニタを操作できます。	• LDAP 接続プールモニタの表示（12.4.1 参照） • LDAP 接続プールの空き待ち監視のリセット（12.4.2 参照） • LDAP 接続の定義情報の表示（12.4.3 参照） • LDAP 接続の障害情報の表示（12.4.4 参照）
JDBC 接続モニタ	J2EE サーバクラスタ内の選択した J2EE サーバの JDBC 接続モニタを操作できます。	• JDBC 接続プールモニタの表示（12.5.1 参照） • JDBC 接続プールの空き待ち監視のリセット（12.5.2 参照） • JDBC 接続の定義情報の表示（12.5.3 参照） • JDBC 接続の障害情報の表示（12.5.4 参照）

12.3　ログインセッションの監視

この節では，［ログインセッションモニタ］画面での操作と，監視できる内容について説明します。

12.3.1　ログインセッションモニタの表示

［ログインセッションモニタ］画面を次の図に示します。

図 12-1　［ログインセッションモニタ］画面

(1)　機能概要

統合ユーザ管理のセッションの状態を参照できます。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

   J2EE サーバの場合

   ［J2EE サーバ］－［J2EE サーバ名］をクリックします。

   J2EE サーバクラスタの場合

   ［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。
4. ［ログインセッションモニタ］をクリックします。

(3) 操作手順

画面での操作はありません。

(4) 画面詳細

画面に表示される項目およびアンカーについて説明します。

セッション数

ログインしている統合ユーザ管理のセッション数が表示されます。

セッション ID

ログインしているセッションのセッション ID が表示されます。

レルム名

ログインしているセッション内のレルム名が表示されます。カスタムログインモジュールでログインしている場合は，JAAS のコンフィグレーションで指定したレルム名になります。

ユーザ ID

ログインしているセッション内のユーザ ID が表示されます。

ログイン時刻

ユーザがログインしたときの時刻が表示されます。

［全停止］アンカー

選択した J2EE サーバ内で稼働中の統合ユーザ管理のセッションを，すべて停止します。停止後，画面を最新の状態に更新します。

なお，稼働中のセッションがない場合は，このアンカーは表示されません。

［停止］アンカー

このアンカーのある行のセッションを停止します。停止後，画面を最新の状態に更新します。

［最新の情報に更新］アンカー

画面を最新の情報に更新します。

画面は時間の経過とともに変わります。常に最新の状態でご参照ください。更新間隔を変更したい場合は，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「7.4.1　画面自動更新の設定」を参照してください。

(5) 注意事項

統合ユーザ管理のセッションフェイルオーバ機能を使用している場合，現在ログインしているユーザのフェイルオーバが発生すると，一時的にフェイルオーバ元とフェイルオーバ先の両方のログインセッションモニタに，同じユーザが表示される場合があります。

また，ログインセッションモニタから統合ユーザ管理のセッションを停止しても，統合ユーザ管理のセッションを停止できません。ログインセッションモニタから統合ユーザ管理のセッションを停止した場合，ユーザが該当セッションにリクエストを送信すると，統合ユーザ管理のセッションが自動的に回復されます。

12.3.2　統合ユーザ管理のセッションの停止

(1)　機能概要

選択した統合ユーザ管理のセッションを停止できます。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

   J2EE サーバの場合

   ［J2EE サーバ］－［J2EE サーバ名］をクリックします。

   J2EE サーバクラスタの場合

   ［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。
4. ［ログインセッションモニタ］をクリックします。

(3)　操作手順

画面での操作手順を次に示します。

1. ［全停止］アンカーまたは［停止］アンカーをクリックします。

(4)　画面詳細

画面に表示されるアンカーについて説明します。

［全停止］アンカー

選択した J2EE サーバ内で稼働中の統合ユーザ管理のセッションを，すべて停止します。停止後，画面を最新の状態に更新します。

なお，稼働中のセッションがない場合は，このアンカーは表示されません。

［停止］アンカー

アンカーのある行のセッションを停止します。停止後，画面を最新の状態に更新します。

画面のこのほかの項目の意味については，「12.3.1　ログインセッションモニタの表示」を参照してください。

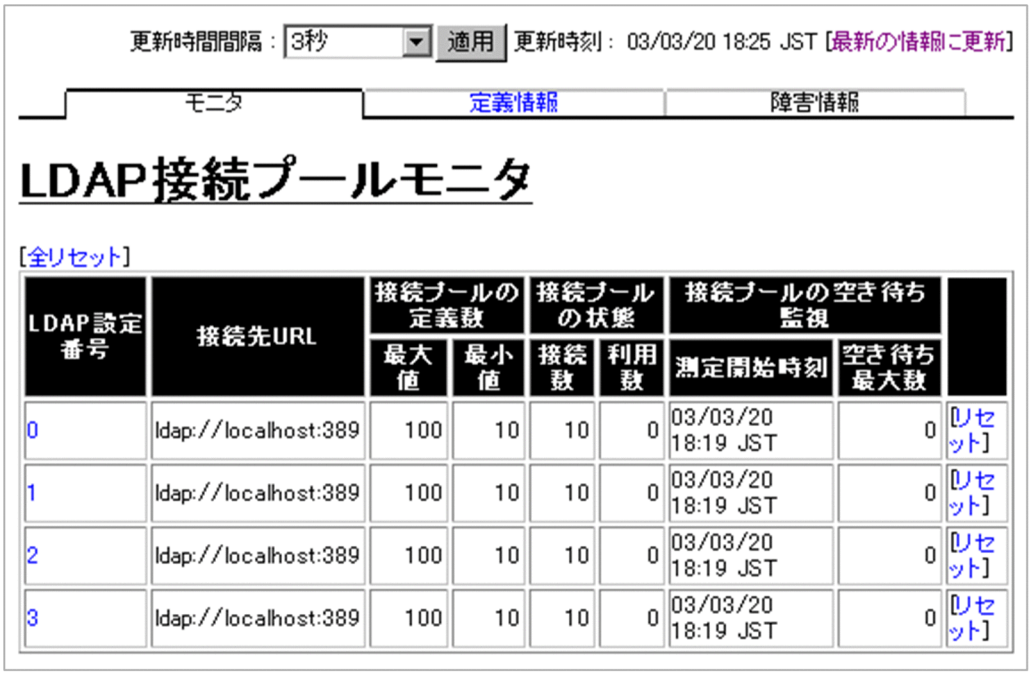
12.4　LDAP 接続モニタの監視

この節では，［LDAP 接続モニタ］画面での操作と，監視できる内容について説明します。

12.4.1　LDAP 接続プールモニタの表示

［LDAP 接続プールモニタ］画面を次の図に示します。

図 12-2　［LDAP 接続プールモニタ］画面



LDAP設定番号	接続先URL	接続プールの定義数		接続プールの状態		接続プールの空き待ち監視		
		最大値	最小値	接続数	利用数	測定開始時刻	空き待ち最大数	
0	ldap://localhost:389	100	10	10	0	03/03/20 18:19 JST	0	[リセット]
1	ldap://localhost:389	100	10	10	0	03/03/20 18:19 JST	0	[リセット]
2	ldap://localhost:389	100	10	10	0	03/03/20 18:19 JST	0	[リセット]
3	ldap://localhost:389	100	10	10	0	03/03/20 18:19 JST	0	[リセット]

(1)　機能概要

統合ユーザ管理が使用する LDAP 接続プールの状態を参照できます。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブー［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

   J2EE サーバの場合

   ［J2EE サーバ］－［J2EE サーバ名］をクリックします。

J2EE サーバクラスタの場合

［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。

4. ［LDAP 接続モニタ］をクリックします。

5. ［モニタ］タブをクリックします。

(3) 操作手順

画面での操作はありません。

(4) 画面詳細

画面に表示される項目およびアンカーについて説明します。

LDAP 設定番号

統合ユーザ管理のコンフィグレーションファイルで指定した，LDAP ディレクトリサーバと接続するための情報を識別する LDAP 設定番号が表示されます。番号のアンカーをクリックすると，詳細情報を参照できます。表示される画面については，「12.4.3　LDAP 接続の定義情報の表示」を参照してください。

接続先 URL

接続先 LDAP ディレクトリサーバの URL が表示されます。定義されていない場合は「未定義」と表示されます。

接続プールの定義数

最大値

LDAP 接続プールの最大数が表示されます。プールを利用しない場合は「－」が表示されます。

最小値

LDAP 接続プールの空きプールの数が 0 になったとき（初期化時を含む），新たに確立されるプール数が表示されます。プールを利用しない場合は「－」が表示されます。

接続プールの状態

接続数

接続している LDAP 接続プールの数が表示されます。LDAP ディレクトリサーバと接続できない場合は「X」が表示されます。LDAP ディレクトリサーバと接続できて，プールを利用しない場合は「－」が表示されます。

「X」のアンカーをクリックすると，障害情報を参照できます。表示される画面については，「12.4.4 LDAP 接続の障害情報の表示」を参照してください。

利用数

現在使用している LDAP 接続プールのプール数が表示されます。プールを利用しない場合は「－」が表示されます。

接続プールの空き待ち監視

測定開始時刻

LDAP 接続プールの空き待ち数のカウントを開始した時刻が表示されます。プールを利用しない場合は「－」が表示されます。

空き待ち最大数

LDAP 接続プールの定義数の最大値を超えて要求があった，LDAP 接続プールの空き待ち数が表示されます。この値は［測定開始時刻］が表示された時刻からの値です。プールを利用しない場合は「－」が表示されます。

［全リセット］アンカー

すべての［接続プールの空き待ち監視］をリセットします。

なお，すべてがプールを使用していない場合は，このアンカーは表示されません。

［リセット］アンカー

このアンカーのある行の［接続プールの空き待ち監視］をリセットします。プールを使用していない行には，このアンカーは表示されません。

［最新の情報に更新］アンカー

画面を最新の情報に更新します。

画面は時間の経過とともに変わります。常に最新の状態でご参照ください。更新間隔を変更したい場合は，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「7.4.1　画面自動更新の設定」を参照してください。

12.4.2　LDAP 接続プールの空き待ち監視のリセット

(1)　機能概要

統合ユーザ管理が使用する LDAP 接続プールで，接続プールの空き待ち監視の設定をリセットできます。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

J2EE サーバの場合

［J2EE サーバ］－［J2EE サーバ名］をクリックします。

J2EE サーバクラスタの場合

［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。

4. [LDAP 接続モニタ] をクリックします。

5. [モニタ] タブをクリックします。

(3) 操作手順

画面での操作手順を次に示します。

1. [全リセット] アンカーまたは [リセット] アンカーをクリックします。

(4) 画面詳細

画面に表示されるアンカーについて説明します。

[全リセット] アンカー

すべての [接続プールの空き待ち監視] をリセットします。

なお，すべてがプールを使用していない場合は，このアンカーは表示されません。

[リセット] アンカー

このアンカーのある行の [接続プールの空き待ち監視] をリセットします。プールを使用していない行には，このアンカーは表示されません。

画面のこのほかの項目の意味については，「12.4.1 LDAP 接続プールモニタの表示」を参照してください。

12.4.3 LDAP 接続の定義情報の表示

[LDAP 設定番号の詳細定義情報] 画面を次の図に示します。

図 12-3 ［LDAP 設定番号の詳細定義情報］画面

項目	値
使用ファクトリクラス	com.sun.jndi.ldap.LdapCtxFactory
接続先URL	ldap://localhost:389
ベースDN	ou=users,ou=RealmA,ou=examples,o=cosminexus.com
バインドDN	cn=Directory Manager
バインドDNのパスワード	********
LDAP接続失敗時のリトライ回数	1
LDAP接続失敗時のリトライ時間間隔	0ミリ秒
ユーザエントリ内でユーザIDを示す属性名	uid
ユーザエントリ内でパスワードを示す属性名	userPassword
証明書からユーザIDを切り出す属性名	cn
接続プールの使用	する

(1) 機能概要

選択した J2EE サーバで定義している，統合ユーザ管理の LDAP 接続情報を参照できます。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

   J2EE サーバの場合

   ［J2EE サーバ］－［J2EE サーバ名］をクリックします。

   J2EE サーバクラスタの場合

   ［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。
4. ［LDAP 接続モニタ］をクリックします。
5. ［定義情報］タブをクリックします。

(3) 操作手順

画面での操作はありません。

(4) 画面詳細

画面に表示される項目およびアンカーについて説明します。

使用ファクトリクラス

JNDI で使用するファクトリクラスの名称が表示されます。

接続先 URL

接続先 LDAP ディレクトリサーバの URL が表示されます。定義されていない場合は「未定義」と表示されます。

ベース DN

ベース DN が表示されます。定義されていない場合は「未定義」と表示されます。

バインド DN

コンフィグレーションファイルに定義されているバインド DN が表示されます。定義されていない場合は「未定義」と表示されます。

バインド DN のパスワード

コンフィグレーションファイルに定義されているバインド DN のパスワードが「********」で表示されます。定義されていない場合は「未定義」と表示されます。

LDAP 接続失敗時のリトライ回数

LDAP 接続に失敗した場合のリトライ回数が表示されます。

LDAP 接続失敗時のリトライ時間間隔

LDAP 接続に失敗した場合のリトライを繰り返すときの待ち時間がミリ秒で表示されます。

ユーザエントリ内でユーザ ID を示す属性名

ユーザエントリ内でユーザ ID として使用する属性名が表示されます。

ユーザエントリ内でパスワードを示す属性名

ユーザエントリ内でパスワードとして使用する属性名が表示されます。

証明書からユーザ ID を切り出す属性名

証明書内に格納されている DN を分解したあと，ユーザ ID として用いる属性名が表示されます。ユーザ ID を取り出すときに同じ属性名が複数存在していた場合は，最初に見つかった値が表示されます。

接続プールの使用

LDAP 接続プールを利用するかどうかが表示されます。

する：LDAP 接続プールを利用します。

しない：LDAP 接続プールを利用しません。

LDAP 接続プールの最大値

LDAP 接続プールの最大数が表示されます。

［接続プールの使用］で「しない」が表示されている場合は，この項目は表示されません。

LDAP 接続プールの空きプール数の最大値

LDAP 接続プールの空きプール数の最大値が表示されます。

［接続プールの使用］で「しない」が表示されている場合は，この項目は表示されません。

LDAP 接続プールの空きプール数の最小値

LDAP 接続プールの空きプールの数が 0 になったとき（初期化時を含む），新たに確立されるプール数が表示されます。

［接続プールの使用］で「しない」が表示されている場合は，この項目は表示されません。

プール数を調整する時間間隔

LDAP 接続プールの空きプール数を調整する時間間隔が，秒単位で表示されます。

なし：プール数の調整は行いません。

1 以上：表示された時間でプール数の調整を行います。

［接続プールの使用］で「しない」が表示されている場合は，この項目は表示されません。

検索の有無

ユーザ情報リポジトリのユーザエントリの検索をするかしないかが表示されます。

する：ユーザエントリを検索します。

しない：ユーザエントリを検索しないで，直接エントリを参照します。

検索時のスコープ

ユーザ情報リポジトリのユーザエントリを検索する場合（「する」の場合）の検索レベルが表示されます。

onelevel：ベース DN 直下のエントリからユーザエントリを検索します。

subtree：ベース DN 以下のエントリからユーザエントリを検索します。

［検索の有無］で「しない」が表示されている場合は，この項目は表示されません。

パスワードの暗号化方式

リポジトリに格納されているパスワードの形式が表示されます。

sha1：SHA-1 形式で格納されている場合に表示されます。

sha224：SHA-224 形式で格納されている場合に表示されます。

sha256：SHA-256 形式で格納されている場合に表示されます。

sha384：SHA-384 形式で格納されている場合に表示されます。

sha512：SHA-512 形式で格納されている場合に表示されます。

md5：MD5 形式で格納されている場合に表示されます。

none：平文で格納されている場合に表示されます。

パスワードの暗号化拡張方式

パスワードのフォーマットが標準で用意されているもの以外の形式の場合に，パスワードを変換するためのクラスが表示されます。

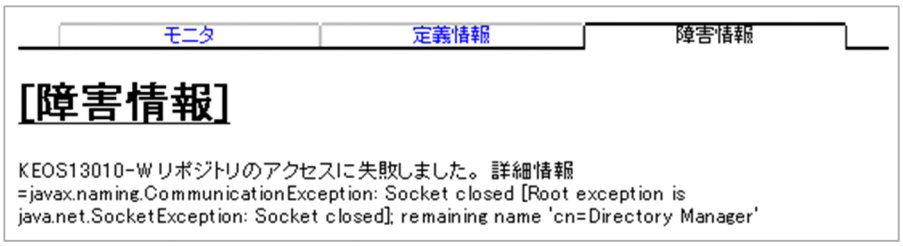
この項目が表示されている場合，［パスワードの暗号化方式］は表示されません。

12.4.4　LDAP 接続の障害情報の表示

ここでは，「リソース監視」で出力される LDAP 接続の障害情報を表示する操作と，障害情報の内容について説明します。

［障害情報］画面を次の図に示します。

図 12–4　［障害情報］画面（例）



(1)　機能概要

統合ユーザ管理の「リソース監視」で発生した現象と，その詳細情報（例外メッセージ）を参照できます。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

   J2EE サーバの場合

   ［J2EE サーバ］－［J2EE サーバ名］をクリックします。

   J2EE サーバクラスタの場合

   ［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。
4. ［LDAP 接続モニタ］をクリックします。
5. ［障害情報］タブをクリックします。

(3)　操作手順

画面での操作はありません。

(4) 画面詳細

画面に表示されるメッセージの要因，対処については，マニュアル「アプリケーションサーバ メッセージ(構築／運用／開発用)」の「13. KEOS（Manager を使用した構築・運用・保守で出力されるメッセージ)」を参照してください。

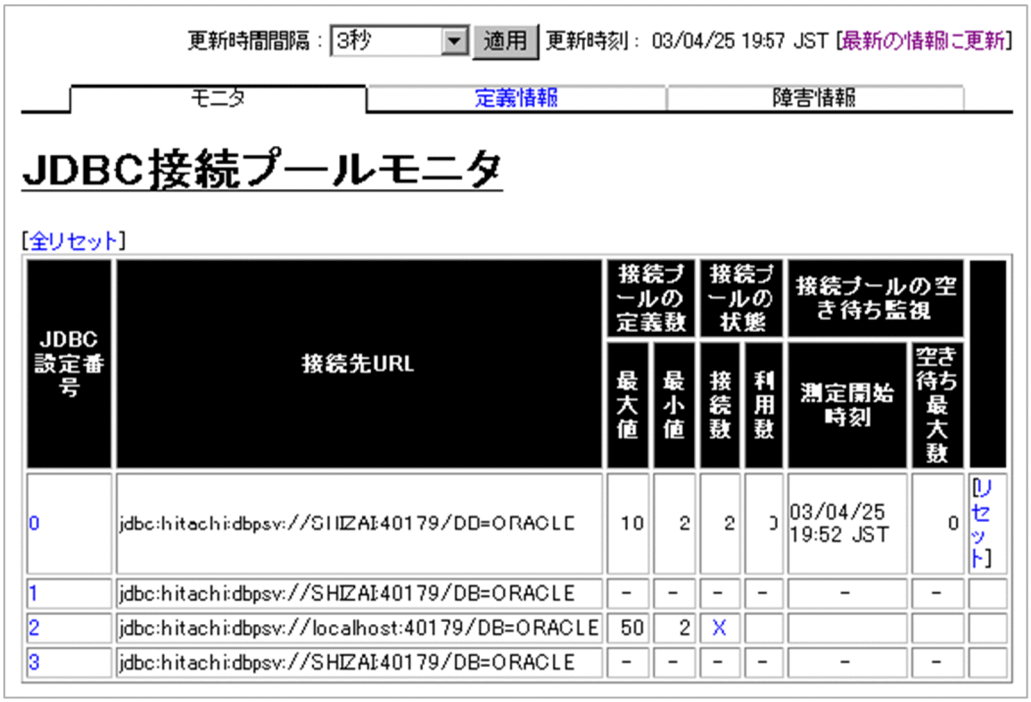
12.5　JDBC 接続モニタの監視

この節では，［JDBC 接続モニタ］画面での操作と，監視できる内容について説明します。

12.5.1　JDBC 接続プールモニタの表示

［JDBC 接続プールモニタ］画面を次の図に示します。

図 12–5　［JDBC 接続プールモニタ］画面



(1)　機能概要

統合ユーザ管理が使用する JDBC 接続プールの状態を参照できます。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

   J2EE サーバの場合

   ［J2EE サーバ］－［J2EE サーバ名］をクリックします。

J2EE サーバクラスタの場合

［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。

4. ［JDBC 接続モニタ］をクリックします。

5. ［モニタ］タブをクリックします。

(3) 操作手順

画面での操作はありません。

(4) 画面詳細

画面に表示される項目およびアンカーについて説明します。

JDBC 設定番号

統合ユーザ管理のコンフィグレーションファイルで指定した，RDB サーバと接続するための情報を識別する JDBC 設定番号が表示されます。番号のアンカーをクリックすると，詳細情報を参照できます。表示される画面については，「12.5.3　JDBC 接続の定義情報の表示」を参照してください。

接続先 URL

接続先 RDB サーバの URL が表示されます。定義されていない場合は「未定義」と表示されます。

接続プールの定義数

最大値

JDBC 接続プールの最大数が表示されます。プールを利用しない場合は「－」が表示されます。

最小値

JDBC 接続プールの空きプールの数が 0 になったとき（初期化時を含む），新たに確立されるプール数が表示されます。プールを利用しない場合は「－」が表示されます。

接続プールの状態

接続数

接続している JDBC 接続プールの数が表示されます。RDB サーバと接続できない場合は「X」が表示されます。RDB サーバと接続できて，プールを利用しない場合は「－」が表示されます。

「X」のアンカーをクリックすると，障害情報を参照できます。表示される画面については，「12.5.4 JDBC 接続の障害情報の表示」を参照してください。

利用数

現在使用している JDBC 接続プールのプール数が表示されます。プールを利用しない場合は「－」が表示されます。

接続プールの空き待ち監視

測定開始時刻

JDBC 接続プールの空き待ち数のカウントを開始した時刻が表示されます。プールを利用しない場合は「－」が表示されます。

空き待ち最大数

JDBC 接続プールの定義数の最大値を超えて要求があった，JDBC 接続プールの空き待ち数が表示されます。この値は［測定開始時刻］が表示された時刻からの値です。プールを利用しない場合は「－」が表示されます。

［全リセット］アンカー

すべての［接続プールの空き待ち監視］をリセットします。

なお，すべてがプールを使用しない場合は，このアンカーは表示されません。

［リセット］アンカー

このアンカーのある行の［接続プールの空き待ち監視］をリセットします。プールを使用していない行には，このアンカーは表示されません。

［最新の情報に更新］アンカー

画面を最新の情報に更新します。

画面は時間の経過とともに変わります。常に最新の状態でご参照ください。更新間隔を変更したい場合は，マニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」の「7.4.1　画面自動更新の設定」を参照してください。

12.5.2　JDBC 接続プールの空き待ち監視のリセット

(1)　機能概要

統合ユーザ管理が使用する JDBC 接続プールで，接続プールの空き待ち監視の設定をリセットできます。

(2)　表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

J2EE サーバの場合

［J2EE サーバ］－［J2EE サーバ名］をクリックします。

J2EE サーバクラスタの場合

［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。

4. ［JDBC 接続モニタ］をクリックします。

5. ［モニタ］タブをクリックします。

(3)　操作手順

画面での操作手順を次に示します。

1. ［全リセット］アンカーまたは［リセット］アンカーをクリックします。

(4)　画面詳細

画面に表示されるアンカーについて説明します。

［全リセット］アンカー

すべての［接続プールの空き待ち監視］をリセットします。

なお，すべてがプールを使用しない場合は，このアンカーは表示されません。

［リセット］アンカー

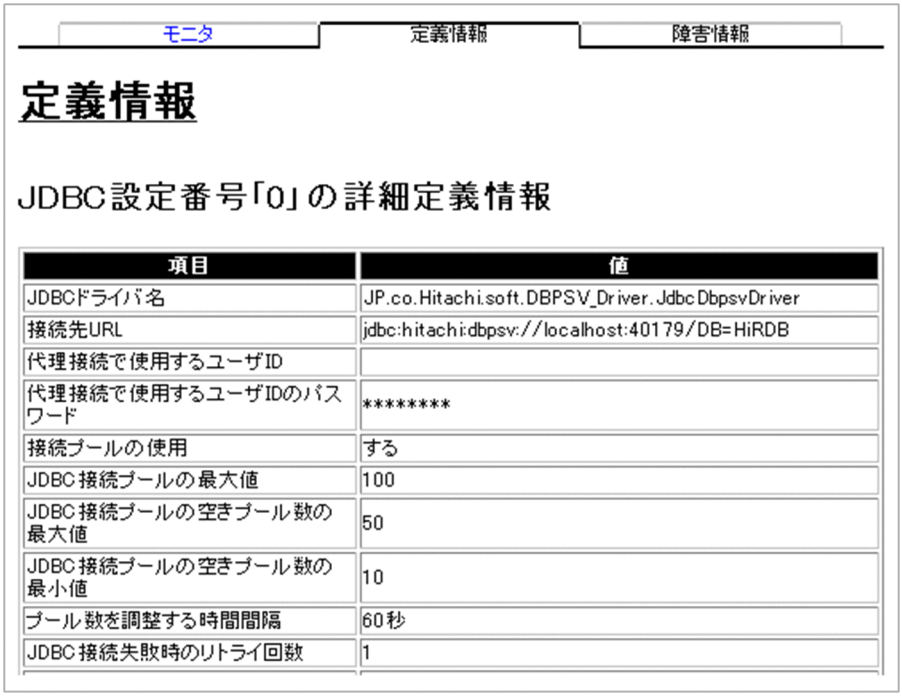
このアンカーのある行の［接続プールの空き待ち監視］をリセットします。プールを使用していない行には，このアンカーは表示されません。

画面のこのほかの項目の意味については，「12.5.1　JDBC 接続プールモニタの表示」を参照してください。

12.5.3　JDBC 接続の定義情報の表示

［JDBC 設定番号の詳細定義情報］画面を次の図に示します。

図 12-6 ［JDBC 設定番号の詳細定義情報］画面



項目	値
JDBCドライバ名	JP.co.Hitachi.soft.DBPSV_Driver.JdbcDbpsvDriver
接続先URL	jdbc:hitachi:dbpsv://localhost:40179/DB=HiRDB
代理接続で使用するユーザID	
代理接続で使用するユーザIDのパスワード	********
接続プールの使用	する
JDBC接続プールの最大値	100
JDBC接続プールの空きプール数の最大値	50
JDBC接続プールの空きプール数の最小値	10
プール数を調整する時間間隔	60秒
JDBC接続失敗時のリトライ回数	1

(1) 機能概要

選択した J2EE サーバで定義している，統合ユーザ管理の JDBC 接続情報を参照できます。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

   J2EE サーバの場合

   ［J2EE サーバ］－［J2EE サーバ名］をクリックします。

   J2EE サーバクラスタの場合

   ［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。
4. ［JDBC 接続モニタ］をクリックします。
5. ［定義情報］タブをクリックします。

(3) 操作手順

画面での操作はありません。

(4) 画面詳細

画面に表示される項目およびアンカーについて説明します。

JDBC ドライバ名

使用する JDBC ドライバ名が表示されます。

「JP.co.Hitachi.soft.DBPSV_Driver.JdbcDbpsvDriver」が表示されます。

接続先 URL

接続先 RDB サーバの URL が表示されます。定義されていない場合は「未定義」と表示されます。

代理接続で使用するユーザ ID

コンフィグレーションファイルに定義されている代理接続で使用するユーザ ID が表示されます。定義されていない場合は「未定義」と表示されます。

代理接続で使用するユーザ ID のパスワード

コンフィグレーションファイルに定義されている代理接続で使用するパスワードが「********」で表示されます。定義されていない場合は「未定義」と表示されます。

接続プールの使用

JDBC 接続プールを利用するかどうかが表示されます。

する：JDBC 接続プールを利用します。

しない：JDBC 接続プールを利用しません。

JDBC 接続プールの最大値

JDBC 接続プールの最大数が表示されます。

［接続プールの使用］で「しない」が表示されている場合は，この項目は表示されません。

JDBC 接続プールの空きプール数の最大値

JDBC 接続プールの空きプール数の最大値が表示されます。

［接続プールの使用］で「しない」が表示されている場合は，この項目は表示されません。

JDBC 接続プールの空きプール数の最小値

JDBC 接続プールの空きプールの数が 0 になったとき（初期化時を含む），新たに確立されるプール数が表示されます。

［接続プールの使用］で「しない」が表示されている場合は，この項目は表示されません。

プール数を調整する時間間隔

JDBC 接続プールの空きプール数を調整する時間間隔が秒単位で表示されます。

なし：プール数の調整は行いません。

1 以上：表示された時間でプール数の調整を行います。

［接続プールの使用］で「しない」が表示されている場合は，この項目は表示されません。

JDBC 接続失敗時のリトライ回数

JDBC 接続に失敗した場合のリトライ回数が表示されます。

JDBC 接続失敗時のリトライ時間間隔

JDBC 接続に失敗した場合のリトライを繰り返すときの待ち時間がミリ秒で表示されます。

SQL 文

パスワードを検索するための SQL SELECT 文が表示されます。定義されていない場合は「未定義」と表示されます。

パスワードの型名

パスワードが格納されている列の値の型が表示されます。型名は，Java 言語で扱う場合の型が次に示す値で表示されます。

string：RDB から String 型でパスワードの値を取り出します。

byte：RDB から byte［］型でパスワードの値を取り出します。

パスワードの暗号化方式

リポジトリに格納されているパスワードの形式が表示されます。

sha1：SHA-1 形式で格納されている場合に表示されます。

sha224：SHA-224 形式で格納されている場合に表示されます。

sha256：SHA-256 形式で格納されている場合に表示されます。

sha384：SHA-384 形式で格納されている場合に表示されます。

sha512：SHA-512 形式で格納されている場合に表示されます。

md5：MD5 形式で格納されている場合に表示されます。

none：平文で格納されている場合に表示されます。

パスワードの暗号化拡張方式

パスワードのフォーマットが標準で用意されているもの以外の形式の場合にパスワードを変換するためのクラスが表示されます。

この項目が表示されている場合，［パスワードの暗号化方式］は表示されません。

12.5.4　JDBC 接続の障害情報の表示

ここでは，「リソース監視」で出力される JDBC 接続の障害情報を表示する操作と，障害情報の内容について説明します。

［障害情報］画面を次の図に示します。

図 12–7　［障害情報］画面（例）

(1) 機能概要

統合ユーザ管理の「リソース監視」で発生した現象と，その詳細情報（例外メッセージ）を参照できます。

(2) 表示手順

画面の表示手順を次に示します。

1. 運用管理ポータルで，統合ユーザ管理の［リソース監視］アンカーをクリックします。
2. ［サーバビュー］タブ－［論理 J2EE サーバ］をクリックします。
3. 次のどちらかの操作をします。

   J2EE サーバの場合

   ［J2EE サーバ］－［J2EE サーバ名］をクリックします。

   J2EE サーバクラスタの場合

   ［J2EE サーバクラスタ］－［J2EE サーバクラスタ名］－［J2EE サーバ名］をクリックします。
4. ［JDBC 接続モニタ］をクリックします。
5. ［障害情報］タブをクリックします。

(3) 操作手順

画面での操作はありません。

(4) 画面詳細

画面に表示されるメッセージの要因，対処については，マニュアル「アプリケーションサーバ メッセージ(構築／運用／開発用)」の「13. KEOS（Manager を使用した構築・運用・保守で出力されるメッセージ)」を参照してください。

13 統合ユーザ管理で使用するコマンド

この章では，統合ユーザ管理で使用するコマンドの入力形式，機能などについて説明します。

13.1　統合ユーザ管理で使用するコマンドの一覧

統合ユーザ管理で使用するコマンドの一覧を，次の表に示します。

表 13–1　統合ユーザ管理で使用するコマンドの一覧

コマンド名称	分類	概要
convpw	パスワードの暗号化	ldif ファイル内のパスワードフィールドの内容を暗号化して，その結果を標準出力に出力します。
ssoexport	シングルサインオン情報リポジトリの参照	シングルサインオン情報リポジトリに内容を読み込んで，標準出力に CSV 形式で表示します。
ssogenkey	暗号鍵ファイルの作成	シングルサインオン情報リポジトリに登録／参照するときの暗号鍵ファイルを作成します。
ssoimport	シングルサインオン情報リポジトリの登録	シングルサインオン用の認証情報が格納されている CSV 形式のファイルを読み込んでシングルサインオン情報リポジトリに登録します。
uachpw	パスワードの変更	統合ユーザ管理のコンフィグレーションファイル中の LDAP ディレクトリサーバと DB サーバにアクセスするためのパスワードを変更します。

13.2　統合ユーザ管理で使用するコマンドの詳細

統合ユーザ管理で使用する各コマンドの入力形式，機能などを次に示します。

コマンドの格納先

統合ユーザ管理で使用するコマンドは，次のディレクトリに格納されています。

- Windows の場合
  <Application Server のインストールディレクトリ>¥manager¥bin¥
- UNIX の場合
  /opt/Cosminexus/manager/bin/

共通の仕様

終了コード

統合ユーザ管理で使用するコマンドの終了コードを，次の表に示します。

表 13-2　統合ユーザ管理で使用するコマンドの終了コード

終了コード	意味
0	正常終了しました。
1	コマンドの実行中にエラーが発生しました。
2	コマンドまたはサーバ起動時の引数に誤りがあります。

convpw（パスワードの暗号化）

形式

```
convpw [-f {md5|sha1|sha224|sha256|sha384|sha512}] <ldif_file_name> <password_attribute>
```

機能

ユーザ情報リポジトリに登録するときに，ldif ファイルの内容を暗号化します。このコマンドは，指定した ldif ファイルの内容を読み取って，<password_attribute>で指定されている内容を暗号化して標準出力に出力します。ldif ファイルの内容で，<password_attribute>で指定した属性名の値以外は，そのまま標準出力に出力します。

なお，<password_attribute>で指定した属性名がない場合，そのままファイルの内容が標準出力に出力されます。

なお，このコマンドは，root 権限があるユーザ，またはコマンドの実行権限を設定した特定のユーザだけが実行できます。特定のユーザにコマンドの実行権限を設定する方法については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「mngenvsetup（管理グループの設定）」を参照してください。

引数

-f {md5|sha1|sha224|sha256|sha384|sha512}

<ldif_file_name>に指定したファイルの内容の中で，password（パスワード）で指定された値を暗号化する形式を指定します。省略した場合は，sha1 を仮定します。ここで指定する値は，大文字と小文字を区別しません。

- md5
  MD5 形式で暗号化します。
- sha1
  SHA-1 形式で暗号化します。
- sha224
  SHA-224 形式で暗号化します。
- sha256
  SHA-256 形式で暗号化します。
- sha384
  SHA-384 形式で暗号化します。
- sha512
  SHA-512 形式で暗号化します。

<ldif_file_name>

パスワード変換をするユーザ情報が格納された ldif ファイルのファイル名を指定します。

<password_attribute>

パスワードのフィールドの内容を変換する際の属性名を指定します。

注意事項

変換する ldif ファイルに日本語を記述する場合，UTF-8 に変換したあとで，base64 エンコードしてください。LDIF の詳細については，RFC 2849 "The LDAP Data Interchange Format (LDIF) - Technical Specification"を参照してください。

ssoexport（シングルサインオン情報リポジトリの参照）

形式

```
ssoexport [-n <レルム名>] [-u <ユーザID>] [-scramble] <useradmin_configfile>
```

機能

シングルサインオン情報リポジトリに格納されたユーザ情報を，CSV 形式で標準出力に出力します。

シングルサインオン情報リポジトリのユーザ情報を変更する場合，ssoexport コマンドによって出力された情報を CSV 形式ファイルにリダイレクトして，CSV 形式ファイルを編集したあとで，ssoimport コマンドで再度登録処理をします。

ssoexport コマンドでユーザ情報を取り出した場合，項目”SECRETDATA”の実データは復号化されます。

ssoexport コマンドがユーザ情報を CSV 形式に変換して出力するには，ユーザ情報にレルムとユーザ ID が含まれている必要があります。含まれていない場合，出力しないで無視します。

なお，このコマンドは，root 権限があるユーザ，またはコマンドの実行権限を設定した特定のユーザだけが実行できます。特定のユーザにコマンドの実行権限を設定する方法については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「mngenvsetup（管理グループの設定)」を参照してください。

引数

-n <レルム名>

検索するレルム名を指定します。省略した場合，すべてのレルム名が対象となります。

-u <ユーザ ID>

検索するユーザ ID を指定します。ユーザ ID には，ワイルドカード（*）を指定できます。ワイルドカード（*）を指定する場合は，ダブルクォーテーション（"）で囲んでください。

（例）

- -u "*"
  すべてのユーザを取り出します。
- -u "Ta*"
  ユーザ ID が Ta で始まるユーザを取り出します。
- -u "*no"
  ユーザ ID が no で終わるユーザを取り出します。

省略した場合，すべてのユーザ ID が対象となります。

-scramble

パスワード変更コマンド（uachpw）でパスワードをスクランブル化した場合に指定します。

<useradmin_configfile>

統合ユーザ管理のコンフィグレーションファイル（ua.conf）を指定します。

入力例・出力例

レルム名が「RealmA」で，「s9」で始まるユーザを取り出す場合の入力例および出力例を次に示します。

入力例

Windows の場合

```
C:¥>ssoexport -n RealmA -u ”s9*” ”C:¥Program Files¥Hitachi¥Cosminexus¥manager¥config¥u
a.conf”
```

UNIX の場合

```
% ssoexport -n RealmA -u ”s9*” /opt/Cosminexus/manager/config/ua.conf
```

出力例

```
SecurityDomain,USERID,SECRETDATA,PUBLICDATA,LINK_J2EE,LINK_REALMA
RealmA,s981234,abfdef,,
RealmA,s991234,ghijkl,,
```

注意事項

- シングルサインオン情報リポジトリに情報がない場合，または指定したレルム名／ユーザ ID に対応したユーザ情報が取得できない場合，ヘッダだけ出力してこのコマンドは終了します。
- ssoexport コマンドを実行中に LDAP ディレクトリサーバを停止しないでください。エラーメッセージを出力しないで，コマンドが終了することがあります。
- ssoexport コマンドと ssoimport コマンドは，同時に実行しないでください。
- ssoimport コマンドで登録する接続先のユーザ管理を持つアプリケーションについては，整合性を取りません。つまり，シングルサインオン情報リポジトリ内に対応するレルムのエントリ（またはユーザエントリ）がなくてもエラーとなりません。この場合，ssoexport コマンドで情報を参照しても出力されないため注意が必要です。ssoexport コマンドでは，接続先のユーザ管理を持つアプリケーションについては，ベース DN の直下にあるレルムのエントリに対応する値しか表示しません。

ssogenkey（暗号鍵ファイルの作成）

形式

```
ssogenkey <useradmin_configfile>
```

機能

シングルサインオン用の認証情報は，暗号化して保存します。また，参照する際は，復号化して参照します。このコマンドは，暗号化および復号化で使用する暗号鍵を作成します。

なお，このコマンドは，root 権限があるユーザ，またはコマンドの実行権限を設定した特定のユーザだけが実行できます。特定のユーザにコマンドの実行権限を設定する方法については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「mngenvsetup（管理グループの設定）」を参照してください。

引数

<useradmin_configfile>

統合ユーザ管理のコンフィグレーションファイル（ua.conf）を指定します。

注意事項

- 指定したファイルがすでにある場合は，”.n”を付けて，指定したファイルと同一ディレクトリ内にバックアップを取ります。
- このコマンドでは，シングルサインオン情報リポジトリの内容はアクセスしません。すでにシングルサインオン情報リポジトリ内にシングルサインオン用の認証情報を登録している場合は，先にすべての情報を ssoexport で取り出したあと，ssoimport コマンドで登録する必要があります。

ssoimport（シングルサインオン情報リポジトリの登録）

形式

```
ssoimport {-a|-m|-d|-x} [-p] [-scramble] <csvfile_name> <useradmin_configfile>
```

機能

ユーザ管理を持つアプリケーションから取り出した CSV 形式ファイル（または取り出して編集した CSV 形式ファイル）を，シングルサインオン情報リポジトリに登録します。登録時，項目 ID”SECRETDATA”に記述された実データは暗号化されてシングルサインオン情報リポジトリに登録されます。

なお，このコマンドは，root 権限があるユーザ，またはコマンドの実行権限を設定した特定のユーザだけが実行できます。特定のユーザにコマンドの実行権限を設定する方法については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「mngenvsetup（管理グループの設定）」を参照してください。

引数

-a

<csvfile_name>に指定したファイルの内容をシングルサインオン情報リポジトリに追加します。シングルサインオン情報リポジトリ内に追加しようとするユーザのエントリがある場合は，該当ユーザに対する処理はしないで，警告メッセージを出力して次行以降の行の処理を継続します。

-m

<csvfile_name>に指定したファイルの内容でシングルサインオン情報リポジトリを上書きします。シングルサインオン情報リポジトリ内に上書きしようとするユーザのエントリがない場合は，ユーザエントリを追加します。

-d

<csvfile_name>に指定したファイルの内容をシングルサインオン情報リポジトリから削除します。シングルサインオン情報リポジトリ内に削除しようとするユーザのエントリがない場合は，警告メッセージを出力して処理は継続します。

-x

ラインオペレーションの指定に従って，シングルサインオン情報リポジトリの内容を更新します。シングルサインオン用認証情報の CSV 形式ファイルのラインオペレーションについては，「14.3　シングルサインオン用認証情報の CSV 形式ファイル」を参照してください。

-p

追加，変更，または更新したレルム名とユーザ名の一覧を標準出力に出力します。

-scramble

パスワード変更コマンド（uachpw）でパスワードをスクランブル化した場合に指定します。

<csvfile_name>

シングルサインオン情報リポジトリに登録する CSV 形式ファイルを指定します。

<useradmin_configfile>

統合ユーザ管理のコンフィグレーションファイル（ua.conf）を指定します。

入力例

ここでは，登録する CSV 形式ファイルを userdata.csv，シングルサインオン用設定ファイルを”ua.conf”として説明します。

CSV 形式ファイルの内容をシングルサインオン情報リポジトリに追加する場合

```
ssoimport -a userdata.csv ua.conf
```

CSV 形式ファイルの内容をシングルサインオン情報リポジトリから削除する場合

```
ssoimport -d userdata.csv ua.conf
```

CSV 形式ファイル OPERATION 項目の実データに指定したオペレーションに従ってシングルサインオン情報リポジトリに登録する場合

```
ssoimport -x userdata.csv ua.conf
```

出力メッセージ

ssoimport コマンドは，オプションに指定した CSV 形式ファイルから 1 行ずつ取得してシングルサインオン情報リポジトリに登録（変更または削除）します。このとき，-p オプションを指定して実行すると，その操作についての情報を標準出力に出力します。なお，操作時に発生した警告，エラーメッセージについては，標準エラー出力に出力します。

-p オプションを指定しないで実行した場合は，警告，エラーメッセージと「結果」だけを出力します。

出力例を次の図に示します。

図 13-1　ssoimport コマンドの出力例（Windows の場合）

```
c:¥> ssoimport -x -p xxx.csv .¥ssoconf.conf
OPERATION    REALMNAME    USERID                          } タイトル
----------------------------------------
add         Portal      USER1                            )
delete      Portal      USER3                            |
xxxx0010-W Not Found User-ID, line=xxx realm=Portal uid=WWW |
 :                                                       |
 : （中略）                                              } 実行内容
 :                                                       |
modify      Portal      jirou                            )
----------------------------------------
Total:100  ADD:90  MODIFY:8  DELETE:1  WARNING:1         } 結果
c:¥>
```

図 13-2　ssoimport コマンドの出力例（UNIX の場合）

```
% ssoimport -x -p xxx.csv ./ssoconf.conf
OPERATION    REALMNAME    USERID                          } タイトル
----------------------------------------
add         Portal      USER1                            )
delete      Portal      USER3                            |
xxxx0010-W Not Found User-ID, line=xxx realm=Portal uid=WWW |
 :                                                       |
 : （中略）                                              } 実行内容
 :                                                       |
modify      Portal      jirou                            )
----------------------------------------
Total:100  ADD:90  MODIFY:8  DELETE:1  WARNING:1         } 結果
%
```

「OPERATION」「REALMNAME」「USERID」の順に表示します。

実行内容

ヘッダに対応した「OPERATION」「REALMNAME」「USERID」の内容，および操作時に発生した警告／エラーメッセージを表示します。

OPERATION

次の表に示す操作種別のどれかを出力します。

表 13-3　ssoimport コマンドの操作種別

項目	内容
add	情報を追加したことを意味します。
modify	情報を変更（上書き）したことを意味します。
delete	情報を削除したことを意味します。

REALMNAME

操作の対象となるレルム名を出力します。この値は，CSV 形式ファイルの「REALMNAME」項目に指定した値を表示します。

USERID

操作の対象となるユーザ ID を出力します。この値は，CSV 形式ファイルのヘッダ「USERID」項目に指定した値を表示します。

実行結果

次の表に示す実行結果を表示します。

表 13-4　ssoimport コマンドの実行結果

項目	内容
Total	操作の対象となった行数を出力します。
ADD	シングルサインオン情報リポジトリ内に追加したエントリ数を表示します。
MODIFY	シングルサインオン情報リポジトリ内を変更したエントリ数を表示します。
DELETE	シングルサインオン情報リポジトリから削除したエントリ数を表示します。
WARNING	上記の操作をした際に発生した警告メッセージの数を表示します。

注意事項

- 接続先の JAAS 対応ユーザ管理を持つアプリケーションのユーザの情報を削除する場合は，ssoexport で対象とするユーザを取得したあとで，削除したい接続先のユーザ管理を持つアプリケーションのユーザを削除し，-m オプションで更新してください。
- ssoexport コマンドと ssoimport コマンドは，同時に実行しないでください。
- ssoimport コマンドで登録する接続先のユーザ管理を持つアプリケーションについては，整合性を取りません。つまり，シングルサインオン情報リポジトリ内に対応するレルムのエントリ（またはユーザエントリ）がなくてもエラーとなりません。この場合，ssoexport コマンドで情報を参照しても出力されないため注意が必要です。ssoexport コマンドでは，接続先のユーザ管理を持つアプリケーションについては，ベース DN の直下にあるレルムのエントリに対応する値しか表示しません。

uachpw（パスワードの変更）

形式

```
uachpw [-scramble] [-ldap.<n> <パスワード>] [-db.<n> <パスワード>] <useradmin_configfile>
```

機能

統合ユーザ管理のコンフィグレーションファイル（ua.conf）の LDAP ディレクトリサーバと DB サーバにアクセスするためのパスワードを変更します。また，変更時にパスワードをスクランブル化できます。

なお，このコマンドは，root 権限があるユーザ，またはコマンドの実行権限を設定した特定のユーザだけが実行できます。特定のユーザにコマンドの実行権限の設定については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「mngenvsetup（管理グループの設定）」を参照してください。

引数

-scramble

変更するパスワードをスクランブル化する場合に指定します。

-ldap.<n> <パスワード>

<n>で指定したLDAPディレクトリサーバにアクセスするためのパスワードを変更します。<n>には，統合ユーザ管理のコンフィグレーションファイルに定義したLDAP設定番号を指定します。<パスワード>には，変更する新しいパスワードを指定します。

-db.<n> <パスワード>

<n>で指定したDBサーバにアクセスするためのパスワードを変更します。<n>には，統合ユーザ管理のコンフィグレーションファイルに定義したJDBC設定番号を指定します。<パスワード>には，変更する新しいパスワードを指定します。

<useradmin_configfile>

統合ユーザ管理のコンフィグレーションファイル（ua.conf）を指定します。この引数は省略できません。

入力例

統合ユーザ管理のコンフィグレーションファイル（ua.conf）に定義したパスワードを下記のように変更し，-scrambleオプションを使用してスクランブル化する場合の入力例を示します。

- LDAPアクセス情報0のパスワードをdiradminに変更
- LDAPアクセス情報1のパスワードをadministratorに変更
- DBアクセス情報0をtigerに変更

```
% uachpw -scramble -ldap.0 diradmin -ldap.1 administrator -db.0 tiger ua.conf
```

注意事項

- -scrambleオプションでパスワードをスクランブル化する場合は，usrconf.propertiesのcom.cosminexus.admin.auth.passwordScramble.enableにtrueを設定してください。設定しなかった場合，パスワードが復号化されないためLDAPディレクトリサーバやDBにアクセスできません。
- パスワードに使用できる文字数は30文字までです。
- オプションの大文字と小文字は区別されません。
- コマンドを二重で実行しないでください。
- 引数に統合ユーザ管理のコンフィグレーションファイル（ua.conf）以外のファイルは指定しないでください。

14 統合ユーザ管理で使用するファイル

この章では，統合ユーザ管理で使用するファイルの形式，格納先，機能，指定できるオプションなどについて説明します。

14.1　統合ユーザ管理で使用するファイルの一覧

統合ユーザ管理で使用するファイルの一覧を，次の表に示します。

表 14–1　統合ユーザ管理で使用するファイルの一覧

ファイル名	分類	概要	参照先
jaas.conf	JAAS のコンフィグレーションファイル	ユーザ認証ライブラリおよびシングルサインオンライブラリの機能を使用するために必要な設定をします。	14.2.1
ua.conf	統合ユーザ管理のコンフィグレーションファイル	JAAS 対応ユーザ管理，およびシングルサインオンの機能を使用するための設定ファイルです。	14.2.2
（任意）	シングルサインオン用認証情報の CSV 形式ファイル	シングルサインオン用の認証情報を設定します。	14.3

14.2　統合ユーザ管理で使用するファイルの詳細

14.2.1　jaas.conf（JAAS のコンフィグレーションファイル）

(1)　形式

ユーザ認証ライブラリおよびシングルサインオンライブラリの機能を使用するために必要な JAAS のコンフィグレーションファイルです。

次のようにオプションを指定します。

```
Application {
  ログインモジュール名 Flag ModuleOptions;
};
```

(2)　ファイルの格納先

- Windows の場合
  <Application Server のインストールディレクトリ>¥manager¥config¥
- UNIX の場合
  /opt/Cosminexus/manager/config/

(3)　機能

ユーザ認証ライブラリおよびシングルサインオンライブラリの機能を使用するために必要な設定をします。JAAS のコンフィグレーションファイルは，運用前に作成し，各ホストに配布する必要があります。その際，盗聴のおそれがあるため，配布の際には注意が必要です。

(4)　指定できるオプション

オプションの名称と内容を次の表に示します。

オプション名	内容
Application	アプリケーション名を指定します。アプリケーションが特定できる名称を使用することを推奨します。なお，ここで指定した名称は，LoginContext クラスのインスタンス化時に使用されます。 Application Server では，次に示す文字列で始まる名称を使用します。そのため，アプリケーション名にこれらの文字列で始まる名称を指定しないでください。 • jp.co.hitachi.soft • com.hitachi.software • com.cosminexus
ログインモジュール名	使用する認証エンジンを指定します。

オプション名	内容
	次のどれかのログインモジュールを指定してください。 • WebPasswordLoginModule パスワードを使用してユーザを認証する場合 • WebCertificateLoginModule クライアント証明書を使用してユーザを認証する場合 • WebPasswordLDAPLoginModule LDAP ディレクトリサーバの認証機能を使用してユーザを認証する場合 • WebPasswordJDBCLoginModule ユーザ情報リポジトリとしてデータベースを使用する場合 • DelegationLoginModule カスタムログインモジュールを呼び出す場合 • WebSSOLoginModule シングルサインオン機能を使用する場合
Flag	LoginContext が呼び出したログインモジュールの正否によって動作を変えるためのフラグを指定します。指定するフラグの詳細については，JAAS のドキュメントを参照してください。
ModuleOptions	ログインモジュールが実行する際に必要なオプションを指定します。指定するオプションの詳細については，「(5)　WebPasswordLoginModule に指定するオプション」～「(10) WebPasswordLDAPLoginModule に指定するオプション」に示します。

(5)　WebPasswordLoginModule に指定するオプション

WebPasswordLoginModule に指定するオプションとデフォルト値を次に示します。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.ldap.r	「14.2.2(3)　リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。指定する値は，ユーザ情報リポジトリを参照可能な設定を，識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。ここで指定された値は，WebPasswordLoginModule を利用してログインする場合などのリポジトリの参照が必要な機能で使用されます。	0
com.cosminexus.admin.auth.ldap.w	「14.2.2(3)　リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。指定する値は，ユーザ情報リポジトリを更新可能な設定を，識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。ここで指定された値は，パスワードの変更機能などの，リポジトリの内容を管理するための機能で使用されます。	0
com.cosminexus.admin.auth.sso.ldap.w	「14.2.2(3)　リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。このオプションは，PasswordUtil クラスを使ってパスワードを変更するときにシングルサインオン用の認証情報も変更する場合，か	<統合ユーザ管理のコンフィグレーショ

オプション	内容	デフォルト値
	つシングルサインオン用の認証情報変更に LDAP 接続フェールオーバを利用する場合に指定してください。指定する値は，シングルサインオン情報リポジトリを更新可能な設定を，識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。 なお，この値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.sso.ldap.w で指定した値よりも優先されます。	ンファイルで指定した値>
com.cosminexus.admin.auth.realm	認証するレルムを文字列で指定します。	なし
com.cosminexus.admin.auth.keep_password	該当レルムにログインしたユーザのパスワードを統合ユーザ管理のセッションに保持するかどうかを，true または false で指定します。大文字と小文字は区別されません。 true を指定した場合は保持します。false を指定した場合は保持しません。なお，true を指定しても，該当レルムですでにログインしている場合，すでに保持しているパスワードを上書きすることはありません。パスワードを保持しない場合，同一セッションの同一レルムに 2 回目以降 WebPasswordLDAPLoginModule を使用してログインするとき，ユーザ ID とパスワードの入力が必要になります。 なお，ここで指定された値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.keep_password で指定された値よりも優先されます。	<統合ユーザ管理のコンフィグレーションファイルで指定した値>
com.cosminexus.admin.auth.keep_password.encrypt	com.cosminexus.admin.auth.keep_password に true を指定した場合，パスワードを保持するときに暗号化するかどうかを，true または false で指定します。大文字と小文字は区別されません。 true を指定した場合はパスワードを暗号化します。 false を指定した場合は暗号化しません。 なお，ここで指定された値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.keep_password.encrypt で指定された値よりも優先されます。	<統合ユーザ管理のコンフィグレーションファイルで指定した値>
com.cosminexus.admin.auth.gsession.keep_password	統合ユーザ管理のセッションフェイルオーバ機能が有効，かつ com.cosminexus.admin.auth.keep_password に true を指定した場合に，統合ユーザ管理のセッションに保持したパスワードをセッションフェイルオーバの対象にするかどうかを指定します。 true を指定した場合： パスワードをグローバルセッションに保持します。 false を指定した場合： パスワードをグローバルセッションに保持しません。 セッションフェイルオーバをした場合，同一セッションの同一レルムに WebPasswordLDAPLoginModule を使用してログインするとき，2 回目以降はユーザ ID とパスワードの入力が必要になります。 （指定例） `com.cosminexus.admin.auth.gsession.keep_password=true`	<統合ユーザ管理のコンフィグレーションファイルで指定した値>

(6) WebSSOLoginModule に指定するオプション

WebSSOLoginModule に指定するオプションとデフォルト値を次に示します。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.sso	WebSSOLoginModule から呼び出すログインモジュールの識別子を指定します。ここで指定された識別子を基に JAAS 対応ユーザ管理の設定ファイルから必要な情報を読み込みます。 なお，この指定が省略されていた場合は，標準ログインモジュール（WebPasswordLoginModule）を仮定します。	WebPasswordLoginModule
com.cosminexus.admin.auth.sso.ldap.r	「14.2.2(3)　リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。指定する値は，シングルサインオン情報リポジトリを参照可能な設定を，識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。ここで指定された値は，WebSSOLoginModule を利用してシングルサインオンをする場合などのリポジトリの参照が必要な機能で使用されます。 なお，この値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.sso.ldap.r で指定した値よりも優先されます。	<統合ユーザ管理のコンフィグレーションファイルで指定した値>
com.cosminexus.admin.auth.sso.ldap.w	「14.2.2(3)　リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。指定する値は，シングルサインオン情報リポジトリを更新可能な設定を，識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。ここで指定された値は，パスワードの変更機能などの，リポジトリの更新が必要な機能で使用されます。 なお，この値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.sso.ldap.w で指定した値よりも優先されます。	<統合ユーザ管理のコンフィグレーションファイルで指定した値>
com.cosminexus.admin.auth.realm	認証するレルムを，文字列で指定します。	なし

(7) DelegationLoginModule に指定するオプション

DelegationLoginModule に指定するオプションとデフォルト値を次に示します。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.custom.lm	DelegationLoginModule が呼び出すカスタムログインモジュール名（クラス名）を，文字列で指定します。指定は，完全限定名（fully qualified name）で記述してください。	なし
com.cosminexus.admin.auth.realm	認証するレルムを，文字列で指定します。	なし

(8) WebCertificateLoginModule に指定するオプション

WebCertificateLoginModule に指定するオプションとデフォルト値を次に示します。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.ldap.r	「14.2.2(3) リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。指定する値は，ユーザ情報リポジトリを参照可能な設定を，識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。ここで指定された値は，WebCertificateLoginModule を利用してログインする場合などのリポジトリの参照が必要な機能で使用されます。	0
com.cosminexus.admin.auth.realm	認証するレルムを，文字列で指定します。	なし

(9) WebPasswordJDBCLoginModule に指定するオプション

WebPasswordJDBCLoginModule に指定するオプションとデフォルト値を次に示します。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.jdbc.r	「14.2.2(3) リポジトリアクセス用パラメタ」で定義した JDBC 設定番号を指定します。指定する値は，ユーザ情報リポジトリを参照可能な設定を，識別する番号です。ただし，ここで指定する値は""で囲む必要があります。ここで指定された値は，WebPasswordJDBCLoginModule を利用してログインする場合などのリポジトリの参照が必要な機能で使用されます。	0
com.cosminexus.admin.auth.realm	認証するレルムを，文字列で指定します。	なし
com.cosminexus.admin.auth.keep_password	該当レルムにログインしたユーザのパスワードを統合ユーザ管理のセッションに保持するかどうかを true または false で指定します。大文字と小文字は区別されません。 true を指定した場合は保持します。false を指定した場合は保持しません。なお，true を指定しても，該当レルムですでにログインしている場合，すでに保持しているパスワードを上書きすることはありません。パスワードを保持しない場合，同一セッションの同一レルムに 2 回目以降 WebPasswordLDAPLoginModule を使用してログインするとき，ユーザ ID とパスワードの入力が必要になります。 ただし，ここで指定された値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.keep_password で指定された値よりも優先されます。	<統合ユーザ管理のコンフィグレーションファイルで指定した値>
com.cosminexus.admin.auth.keep_password.encrypt	com.cosminexus.admin.auth.keep_password に true を指定した場合，パスワードを保持するときに暗号化するかどうかを true または false で指定します。大文字と小文字は区別されません。 true を指定した場合はパスワードを暗号化します。 false を指定した場合は暗号化しません。	<統合ユーザ管理のコンフィグレーションファイルで指定した値>

オプション	内容	デフォルト値
	ただし，ここで指定された値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.keep_password.encrypt で指定された値よりも優先されます。	
com.cosminexus.admin.auth.gsession.keep_password	統合ユーザ管理のセッションフェイルオーバ機能が有効，かつ com.cosminexus.admin.auth.keep_password に true を指定した場合に，統合ユーザ管理のセッションに保持したパスワードをセッションフェイルオーバの対象にするかどうかを指定します。 true を指定した場合： パスワードをグローバルセッションに保持します。 false を指定した場合： パスワードをグローバルセッションに保持しません。 セッションフェイルオーバをした場合，同一セッションの同一レルムに WebPasswordLDAPLoginModule を使用してログインするとき，2 回目以降はユーザ ID とパスワードの入力が必要になります。 （指定例） `com.cosminexus.admin.auth.gsession.keep_password=true`	<統合ユーザ管理のコンフィグレーションファイルで指定した値>

(10) WebPasswordLDAPLoginModule に指定するオプション

WebPasswordLDAPLoginModule に指定するオプションとデフォルト値を次に示します。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.ldap.r	「14.2.2(3) リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。指定する値は，ユーザ情報リポジトリを，参照可能な設定を識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。ここで指定された値は，WebPasswordLDAPLoginModule を利用してログインする場合などのリポジトリの参照が必要な機能で使用されます。	0
com.cosminexus.admin.auth.ldap.w	「14.2.2(3) リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。指定する値は，ユーザ情報リポジトリを，更新可能な設定を識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。ここで指定された値は，パスワードの変更機能などの，リポジトリの内容を管理するための機能で使用されます。	0
com.cosminexus.admin.auth.sso.ldap.w	「14.2.2(3) リポジトリアクセス用パラメタ」で定義した LDAP 設定番号を指定します。このオプションは，PasswordUtil クラスを使ってパスワードを変更するときにシングルサインオン用の認証情報も変更する場合，かつシングルサインオン用の認証情報変更に LDAP 接続フェールオーバを利用する場合に指定してください。指定する値は，シングルサインオン情報リポジトリを更新可能な設定を，識別する番号です。ただし，ここで指定する値は""で囲む必要があります。コンマ（,）で区切ることで LDAP 設定	<統合ユーザ管理のコンフィグレーションファイルで指定した値>

オプション	内容	デフォルト値
	番号を複数指定できます。複数指定すると，最初に指定された LDAP ディレクトリサーバがダウンした場合に自動的に切り替えができます。 なお，この値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.sso.ldap.w で指定した値よりも優先されます。	
com.cosminexus.admin.auth.realm	認証するレルムを，文字列指定します。	なし
com.cosminexus.admin.auth.keep_password	該当レルムにログインしたユーザのパスワードを統合ユーザ管理のセッションに保持するかどうかを true または false で指定します。大文字と小文字は区別されません。 true を指定した場合は保持します。false を指定した場合は保持しません。なお，true を指定しても，該当レルムですでにログインしている場合，すでに保持しているパスワードを上書きすることはありません。パスワードを保持しない場合，同一セッションの同一レルムに 2 回目以降 WebPasswordLDAPLoginModule を使用してログインするとき，ユーザ ID とパスワードの入力が必要になります。 なお，ここで指定された値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.keep_password で指定された値よりも優先されます。	<統合ユーザ管理のコンフィグレーションファイルで指定した値>
com.cosminexus.admin.auth.keep_password.encrypt	com.cosminexus.admin.auth.keep_password に true を指定した場合，パスワードを保持するときに暗号化するかどうかを true または false で指定します。大文字と小文字は区別されません。 true を指定した場合はパスワードを暗号化します。 false を指定した場合は暗号化しません。 なお，ここで指定された値は統合ユーザ管理のコンフィグレーションファイルの com.cosminexus.admin.auth.keep_password.encrypt で指定された値よりも優先されます。	<統合ユーザ管理のコンフィグレーションファイルで指定した値>
com.cosminexus.admin.auth.gsession.keep_password	統合ユーザ管理のセッションフェイルオーバ機能が有効，かつ com.cosminexus.admin.auth.keep_password に true を指定した場合に，統合ユーザ管理のセッションに保持したパスワードをセッションフェイルオーバの対象にするかどうかを指定します。 true を指定した場合： パスワードをグローバルセッションに保持します。 false を指定した場合： パスワードをグローバルセッションに保持しません。 セッションフェイルオーバをした場合，同一セッションの同一レルムに WebPasswordLDAPLoginModule を使用してログインするとき，2 回目以降はユーザ ID とパスワードの入力が必要になります。 （指定例） `com.cosminexus.admin.auth.gsession.keep_password=true`	<統合ユーザ管理のコンフィグレーションファイルで指定した値>

14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）

(1)　形式

JAAS 対応ユーザ管理，およびシングルサインオンの機能を使用するための設定ファイルです。

(2)　ファイルの格納先

- Windows の場合
  <Application Server のインストールディレクトリ>¥manager¥config¥
- UNIX の場合
  /opt/Cosminexus/manager/config/

(3)　リポジトリアクセス用パラメタ

JAAS 対応ユーザ管理のリポジトリ（LDAP ディレクトリサーバまたはデータベース）アクセスに関係する情報を定義します。ここで定義する内容は，ログインモジュールや各種コマンドでリポジトリにアクセスする場合に使用されます。

項目は設定番号（LDAP 設定番号または JDBC 設定番号）を付加して複数定義できます。設定番号は 0 から始まり，順番に 1 ずつ加算していきます。もし，数字が飛んだ場合は，定義がそこで切れます。例えば，次に示す例の場合，0 と 1 が定義されたものとみなされます（3 は 2 が指定されていないため無視されます）。

（例）

```
java.naming.provider.url.0=ldap://localhost:389
java.naming.provider.url.1=ldap://localhost:389
java.naming.provider.url.3=ldap://localhost:389
#java.naming.probider.url.3の指定は無視されます。
```

なお，JNDI 用と JDBC 用の定義は別なので，それぞれ「0」から始めます。

(a)　JNDI 用パラメタ

JNDI を使用して LDAP ディレクトリサーバにアクセスする場合に定義する項目です。LDAP ディレクトリサーバにアクセスするための設定は，LDAP 設定番号を 0 から順に上げることで複数指定できます。

オプション	内容	デフォルト値
java.naming.provider.url	リポジトリ（LDAP ディレクトリサーバ）の URL を，文字列で指定します。詳細については，Java の JNDI の説明を参照してください。	なし
java.naming.security.principal	リポジトリ（LDAP ディレクトリサーバ）にアクセスする際の認証者の識別子を，文字列で指定します。詳細については，Java の JNDI の説明を参照してください。	なし
java.naming.security.credentials	java.naming.security.principal.n に対応するパスワードを，文字列で指定します。詳細については，Java の JNDI の説明を参照してください。	なし

オプション	内容	デフォルト値
com.cosminexus.admin.auth.ldap.basedn	JAAS 対応ユーザ管理でのリポジトリのベース DN を，文字列で指定します。	なし
com.cosminexus.admin.auth.ldap.attr.userid	ユーザのログイン ID を表す属性名を，文字列で指定します。	uid
com.cosminexus.admin.auth.ldap.search.userrdn	JAAS 対応ユーザ管理でのリポジトリのユーザエントリ（RDN）を検索する必要があるかどうかを true または false で指定します。ユーザのログイン ID を表す属性名とユーザエントリ（RDN）が異なる場合は，true を指定してください。大文字と小文字は区別されません。	false
com.cosminexus.admin.auth.ldap.search.scope	JAAS 対応ユーザ管理でのリポジトリのユーザエントリ（RDN）を検索する必要がある場合，検索のレベルを onelevel（1 階層下だけ検索）または subtree（すべての下層の検索）で指定します。大文字と小文字は区別されません。	onelevel
com.cosminexus.admin.auth.ldap.attr.password	ユーザのパスワードを表す属性名を，文字列で指定します。	userPassword
com.cosminexus.admin.auth.ldap.pool.enable	LDAP 接続プールを利用するかどうかを true または false で指定します。大文字と小文字は区別されません。	false
com.cosminexus.admin.auth.ldap.pool.max	LDAP 接続プールの最大数を指定します。最大数を超えて要求が来た場合は，プールが空くのを待ちます。0～2147483647 の整数で指定します。0 以下を指定した場合は，100 を仮定します。	100
com.cosminexus.admin.auth.ldap.pool.max_spare	LDAP 接続プールの空きプール数の最大値を指定します。一時的に，指定した最大値を超えることがありますが，com.cosminexus.admin.auth.ldap.pool.gc_interval で指定した時間間隔で調整されます。0～2147483647 の整数で指定します。 ここに指定する値が com.cosminexus.admin.auth.ldap.pool.max よりも大きい値の場合，com.cosminexus.admin.auth.ldap.pool.max で指定した値を仮定します。0 以下の値を指定した場合は，com.cosminexus.admin.auth.ldap.pool.max の 1/2 を仮定します。com.cosminexus.admin.auth.ldap.pool.max に奇数値が指定されていた場合は切り捨てられ，1 の場合は 1 とします。	50
com.cosminexus.admin.auth.ldap.pool.min_spare	LDAP 接続プールの空きプールの数が 0 になったとき（初期化時を含みます），新たに確立されるプール数を指定します。0～2147483647 の整数で指定します。 ここに指定する値が com.cosminexus.admin.auth.ldap.pool.max_spare よりも大きい値の場合，com.cosminexus.admin.auth.ldap.pool.max_spare で指定した値を仮定します。0 以下の値を指定した場合は，com.cosminexus.admin.auth.ldap.pool.max_spare の 1/2 を仮定します。com.cosminexus.admin.auth.ldap.pool.max_spare に奇数値が指定されていた場合は切り捨てられ，1 の場合は 1 とします。	10
com.cosminexus.admin.auth.ldap.pool.gc_interval	LDAP 接続プールの空きプール数を調整する時間間隔を，0～2147483647 の整数（単位：秒）で指定します。com.cosminexus.admin.auth.ldap.pool.max_spare の説明を参照してください。なお，0 以下の値を指定した場合は，この機能は実行されません（com.cosminexus.admin.auth.ldap.pool.max まで増えて削除されません）。	60

オプション	内容	デフォルト値
com.cosminexus.admin.auth.ldap.conn.retry.count	LDAP 接続失敗時のリトライ回数を 0～2147483647 の整数で指定します。	1
com.cosminexus.admin.auth.ldap.conn.retry.wait	LDAP 接続失敗時のリトライ間隔を 0～2147483647 の整数（単位：ミリ秒）で指定します。	0
com.cosminexus.admin.auth.ldap.certificate.attr.userid	証明書に格納されている DN を分解したあと，ユーザ ID として使用する属性名を，文字列で指定します。大文字と小文字は区別されません。ユーザ ID を取り出すときに，同じ属性名が複数あった場合は，最初に見つかった値を使用します。	cn
com.cosminexus.admin.auth.ldap.password.encrypt	リポジトリに格納されているパスワードの形式を指定します。WebPasswordLoginModule では，ここで指定した形式でパスワードを比較します。 • sha1：SHA-1 形式 • sha224：SHA-224 形式 • sha256：SHA-256 形式 • sha384：SHA-384 形式 • sha512：SHA-512 形式 • none：平文 • md5：MD5 形式 ここで指定する文字列は大文字と小文字は区別されません。また，上記以外の文字列を指定した場合は sha1 を仮定します。 com.cosminexus.admin.auth.ldap.password.encrypt.ex を指定した場合，このパラメタは無視されます。	sha1
com.cosminexus.admin.auth.ldap.password.encrypt.ex	パスワードのフォーマットが標準で用意されているもの以外の形式の場合に，パスワードを変換するためのクラスを完全限定名で指定します。 このパラメタを省略した場合，または指定したが見つからない場合， com.cosminexus.admin.auth.ldap.password.encrypt で指定した形式でパスワードを比較します。	なし
com.cosminexus.admin.auth.ldap.directory.kind	接続する LDAP ディレクトリサーバの種類を指定します。 AD： Active Directory を使用する場合に指定します。 ETC： Active Directory 以外の LDAP ディレクトリサーバを使用する場合に指定します。 上記以外の値を指定した場合は，ETC が設定されます。	ETC
com.cosminexus.admin.auth.ldap.conn.read_timeout	LDAP ディレクトリサーバとの読み込みタイムアウト時間を 0～3600 までの整数（単位：秒）で指定します。0 を指定した場合はタイムアウトしません。	3
com.sun.jndi.ldap.connect.timeout	LDAP ディレクトリサーバとの接続タイムアウトを 0 以上の整数（単位：ミリ秒）で指定します。0 以下の整数を指定した場合は，TCP などのネットワークプロトコルのタイムアウト値が使用されます。	LDAP プロバイダの仕様

(b) JDBC 用パラメタ

JDBC を使用してデータベースにアクセスする場合に定義する項目です。データベースと接続するための設定は，JDBC 設定番号の値を 0 から順に上げることで複数指定できます。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.jdbc.driver	使用するデータベースに対応した JDBC ドライバのクラス名を指定します。JDBC ドライバの格納場所は，J2EE サーバのクラスパスに指定してください。	JP.co.Hitachi.soft.DBPSV_Driver.JdbcDbpsvDriver
com.cosminexus.admin.auth.jdbc.conn.url	データベースと接続するための URL を，文字列で指定します。URL は次の形式で指定します。 （指定例） jdbc:<subprotocol>:<subname>	なし
com.cosminexus.admin.auth.jdbc.conn.user	代理で接続するデータベースユーザを，文字列で指定します。省略した場合は，接続するデータベースユーザなしで接続します。	代理者なし
com.cosminexus.admin.auth.jdbc.conn.password	代理で接続するデータベースユーザのパスワードを指定します。com.cosminexus.admin.auth.jdbc.conn.user を省略した場合は無視されます。com.cosminexus.admin.auth.jdbc.conn.user を指定してこのパラメタを省略した場合は，空文字を仮定します。	<空文字>
com.cosminexus.admin.auth.jdbc.pool.enable	JDBC 接続プールを利用するかどうかを true または false で指定します。大文字と小文字は区別されません。	false
com.cosminexus.admin.auth.jdbc.pool.max	JDBC 接続プールの最大数を指定します。最大数を超えて要求が来た場合はプールが空くのを待ちます。0～2147483647 の整数で指定します。0 以下を指定した場合は 100 を仮定します。	100
com.cosminexus.admin.auth.jdbc.pool.max_spare	JDBC 接続プールの空きプール数の最大値を指定します。一時的に，指定した最大値を超えることがありますが，com.cosminexus.admin.auth.jdbc.pool.gc_interval で指定した時間間隔で調整されます。0～2147483647 の整数で指定します。 ここで指定した値が com.cosminexus.admin.auth.jdbc.pool.max よりも大きい値の場合，com.cosminexus.admin.auth.jdbc.pool.max で指定した値を仮定します。0 以下を指定した場合は com.cosminexus.admin.auth.jdbc.pool.max の 1/2 を仮定します。com.cosminexus.admin.auth.jdbc.pool.max に奇数値が指定されていた場合は切り捨てられ，1 の場合は 1 とします。	50
com.cosminexus.admin.auth.jdbc.pool.min_spare	JDBC 接続プールの空きプール数が 0 になったとき（初期化時も含みます），新たに確立されるプール数を指定します。0～2147483647 の整数で指定します。ここで指定した値が com.cosminexus.admin.auth.jdbc.pool.max_spare よりも大きい値の場合，com.cosminexus.admin.auth.jdbc.pool.max_spare で指定した値を仮定します。0 以下を指定した場合は com.cosminexus.admin.auth.jdbc.pool.max_spare の 1/2 を仮定します。com.cosminexus.admin.auth.jdbc.pool.max_spare に奇数値が指定されていた場合は切り捨てられ，1 の場合は 1 とします。	10
com.cosminexus.admin.auth.jdbc.pool.gc_interval	JDBC 接続プールの空きプール数を調整する時間間隔を 0～2147483647 の整数（単位：秒）で指定します。com.cosminexus.admin.auth.jdbc.pool.max_spare の説明を参照してくださ	60

オプション	内容	デフォルト値
	い。0 以下を指定した場合はこの機能は実行されません。 com.cosminexus.admin.auth.jdbc.pool.max の指定値まで増えて削除されません。	
com.cosminexus.admin.auth.jdbc.conn.retry.count	JDBC 接続失敗時のリトライ回数を，0～2147483647 の整数で指定します。	1
com.cosminexus.admin.auth.jdbc.conn.retry.wait	JDBC 接続失敗時のリトライ間隔を，0～2147483647 の整数（単位：ミリ秒）で指定します。	0
com.cosminexus.admin.auth.jdbc.sql	パスワードを検索するための SQL の SELECT 文を指定します。SELECT 文は次の形式で指定してください。 (指定形式) SELECT　列名　FROM　表名　WHERE　検索条件 なお，検索条件には一つだけ’？’IN パラメタプレースホルダーを含められます。この値は，認証時に指定されたユーザ ID に置換されます。	なし
com.cosminexus.admin.auth.jdbc.password.type	パスワードが格納されている列の値の型を指定します。Java 言語で扱う場合の型を，次に示す値で指定します。 • string： データベースから String 型でパスワードの値を取り出します。SQL のデータ型は CHAR/VARCHAR/LONGVARCHAR です。 • byte： データベースから byte[]型でパスワードの値を取り出します。SQL のデータ型は VARBINARY/LONGVARBINARY です。 ここで指定するキーワードは大文字と小文字が区別されません。また，上記のキーワード以外を指定した場合は string を仮定します。	string
com.cosminexus.admin.auth.jdbc.password.encrypt	リポジトリに格納されているパスワードの形式を指定します。WebPasswordLoginModule では，ここで指定した形式でパスワードを比較します。 • sha1：SHA-1 形式 • sha224：SHA-224 形式 • sha256：SHA-256 形式 • sha384：SHA-384 形式 • sha512：SHA-512 形式 • none：平文 • md5：MD5 形式 ここで指定するキーワードは大文字と小文字が区別されません。また，上記のキーワード以外を指定した場合は none を仮定します。 com.cosminexus.admin.auth.jdbc.password.encrypt.ex を指定した場合，このパラメタは無視されます。 このパラメタに sha1 または md5 を指定した場合は，com.cosminexus.admin.auth.jdbc.password.type パラメタに byte を指定してください。	none

オプション	内容	デフォルト値
com.cosminexus.admin.auth.jdbc.password.encrypt.ex	パスワードのフォーマットが標準で用意されているもの以外の形式の場合に，パスワードを変換するためのクラスを完全限定名で指定します。 このパラメタを省略した場合，または指定したが見つからない場合は com.cosminexus.admin.auth.jdbc.password.encrypt で指定した暗号化形式でパスワードを比較します。	なし

(c) API 用パラメタ

LDAP ディレクトリサーバ上のユーザ情報リポジトリを参照または更新する API で使用する情報を定義する項目です。

項目には「.<name>」を付加します。「.<name>」はユーザ情報リポジトリを参照または更新する API を利用するときの識別子として定義します。ここで指定した name は LdapUserDataManager のコンストラクタで指定します。

name の形式

```
<アプリケーションのJavaパッケージ名>.<内部名>
```

内部名：英数字（A～Z，a～z，0～9）またはピリオド（.）で構成される文字列

（例）

```
com.cosminexus.admin.auth.api.repository.ldap.config.<com.cosminexus.admin.auth.Example>=1
```

name は値を変えることで複数定義できます。複数定義する場合，それぞれの name はこのコンフィグレーションファイル内で一意になるようにしてください。また，name は英数字列（A～Z，a～z，0～9）またはピリオド（.）で指定してください。これ以外の文字を使用した場合，誤認識されることがあります。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.api.repository.ldap.config	API で使用する LDAP ディレクトリサーバの定義の識別子（JNDI 用パラメタで指定した LDAP 設定番号）を指定します。	なし

(d) シングルサインオン用パラメタ

シングルサインオン機能を利用する場合に必要な情報です。WebSSOLoginModule が使用する項目と，カスタムログインモジュールを呼び出すための定義項目の，2 種類の情報を指定する必要があります。このうち，カスタムログインモジュールを呼び出すための定義項目（com.cosminexus.admin.auth.sso.lm 以後）には，.name を付加します。name は WebSSOLoginModule から呼び出すログインモジュールの識別子です。

この「.name」を変えることで複数の値が指定できます。ここで指定した「.name」は，JAAS のコンフィグレーションで使用されます。

形式

WebSSOLoginModule が使用する項目=値

カスタムログインモジュールを呼び出すための定義項目.name=値

（例）

```
com.cosminexus.admin.auth.sso.keyfile=d:/tmp/DES3key.key
com.cosminexus.admin.auth.sso.lm.krb5=com.sun.security.module.Krb5LoginModule
com.cosminexus.admin.auth.sso.param.userid.Krb5=javax.security.auth.login.name
…
```

オプション	内容	デフォルト値
com.cosminexus.admin.auth.sso.keyfile	シングルサインオン用の情報を登録する際に暗号化するための鍵情報が格納されているファイル名を絶対パスで指定します。このファイルが指定されていない場合，シングルサインオン機能を利用してログインするときや，パスワードの変更機能（PasswordUtil クラス）で，LoginException 例外が発生します。なお，com.cosminexus.admin.auth.sso.encrypt=none の場合，ここで指定した値は無視されます。	なし
com.cosminexus.admin.auth.sso.encrypt	シングルサインオン用の認証情報（SecretData）を暗号化するために使用する製品を指定します。 • JCE：JCE を使用します。 • NONE：暗号化機能を使用しません。 ここで指定するキーワードは大文字と小文字を区別しません。	NONE
com.cosminexus.admin.auth.sso.ldap.r	この項の先頭で定義した LDAP 設定番号を指定します。指定する値は，シングルサインオン情報リポジトリを参照可能な設定を，識別する番号です。ここで指定された値は，WebSSOLoginModule を利用したシングルサインオンをする場合などの，リポジトリの参照が必要な機能で使用されます。	0
com.cosminexus.admin.auth.sso.ldap.w	この項の先頭で定義した LDAP 設定番号を指定します。指定する値は，シングルサインオン情報リポジトリを更新可能な設定を，識別する番号です。ここで指定された値は，パスワードの変更機能や，SSOExport/SSOImport コマンドなどのリポジトリの内容を管理するための機能で使用します。	0
com.cosminexus.admin.auth.sso.lm	WebSSOLoginModule が呼び出す各アプリケーションのログインモジュール名（クラス名）を指定します。記述する場合は，フルパッケージで記述してください。	なし
com.cosminexus.admin.auth.sso.param.userid	シングルサインオン情報リポジトリに登録されているユーザ ID を渡すためのパラメタ名を指定します。なお，ここで指定された値は，すでに認証済みの場合だけ login()メソッドを呼び出す前に WebSSOLoginModule が設定します。	com.cosminexus.admin.auth.sso.userid
com.cosminexus.admin.auth.sso.param.secdat	シングルサインオン情報リポジトリに登録されている暗号化された情報を渡すためのパラメタ名を指定します。ここで指定したパラメタ名のキー，およびキーに対する値は，login()メソッドを呼び出す前に，WebSSOLoginModule によって設定されます。ただし，キーおよび値が設定されるのは，認証済みの場合だけです。また，値には，復号化されたデータが設定されます。	com.cosminexus.admin.auth.sso.secdat
com.cosminexus.admin.auth.sso.param.pubdat	シングルサインオン情報リポジトリに登録されている暗号化していない情報を渡すためのパラメタ名を指定します。なお，ここで指定された値は，認証済みの場合だけ login()メソッドを呼び出す前に WebSSOLoginModule が設定します。 com.cosminexus.admin.auth.sso.param.userid， com.cosminexus.admin.auth.sso.param.secdat，および	com.cosminexus.admin.auth.sso.pubdat

オプション	内容	デフォルト値
	com.cosminexus.admin.auth.sso.param.pubdat を指定する場合は，パラメタ名が重ならないように指定してください。もし，重なったパラメタを指定した場合，内容の保証はできません。	

(e)　カスタムログインモジュールのパラメタ

DelegationLoginModule または WebSSOLoginModule からカスタムログインモジュールを呼び出すために必要な情報です。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.custom.modules	カスタムログインモジュールおよびカスタムログインモジュールに関連するクラス（Principal や Credintial クラスなど）を格納したディレクトリを絶対パスで指定します。	なし

(f)　標準ログインモジュールのパラメタ

標準ログインモジュール全体に関係する定義情報です。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.keep_password	該当レルムにログインしたユーザのパスワードを統合ユーザ管理のセッションに保持するかどうかを true または false で指定します。大文字と小文字は区別されません。true を指定した場合は保持します。false を指定した場合は保持しません。なお，true を指定しても，該当レルムですでにログインしている場合，すでに保持しているパスワードを上書きすることはありません。パスワードを保持しない場合，同一セッションの同一レルムに 2 回目以降 WebPasswordLDAPLoginModule を使用してログインするとき，ユーザ ID とパスワードの入力が必要になります。	false
com.cosminexus.admin.auth.keep_password.encrypt	com.cosminexus.admin.auth.keep_password に true を指定した場合，パスワードを保持するときに暗号化するかどうかを true または false で指定します。大文字と小文字は区別されません。 true を指定した場合はパスワードを暗号化します。 false を指定した場合は暗号化しません。	true
com.cosminexus.admin.auth.param_check.enable	com.cosminexus.admin.auth.param_check.enable に true を指定した場合，次の標準ログインモジュールを使用してログインユーザ名の前後に空白を入れてログインすると例外が発生します。 • WebPasswordLoginModule • WebPasswordJDBCLoginModule • WebCertificateLoginModule • WebPasswordLDAPLoginModule	true
com.cosminexus.admin.auth.gsession.keep_password	統合ユーザ管理のセッションフェイルオーバ機能が有効な場合，かつ，com.cosminexus.admin.auth.keep_password に true を指定した場合に，統合ユーザ管理のセッションに保持したパスワードをセッションフェイルオーバの対象にするかどうかを指定します。	false

オプション	内容	デフォルト値
	true を指定した場合： パスワードをグローバルセッションに保持します。 false を指定した場合： パスワードをグローバルセッションに保持しません。 セッションフェイルオーバをした場合，同一セッションの同一レルムに WebPasswordLDAPLoginModule を使用してログインするとき，2 回目以降はユーザ ID とパスワードの入力が必要になります。 （指定例） `com.cosminexus.admin.auth.gsession.keep_password=true`	

(g) そのほかのパラメタ

標準ログインモジュールによるユーザ管理全体に関係する定義情報としてトレースファイルがあります。

オプション	内容	デフォルト値
com.cosminexus.admin.auth.trace.prefix	フルパスを含んだトレースファイル名を指定します（拡張子は付けません）。ここに指定した値に”.n.log”という拡張子を付加して出力します（n は 1～面数（最大 16））。この指定がない場合は，トレースログは出力しません。	なし
com.cosminexus.admin.auth.trace.level	トレースレベルを数字で指定します。指定したレベル以内でトレース情報を出力します。 0： ログインおよびログアウトで失敗した場合，トレースログを出力します。 5： ログインおよびログアウトで成功した場合とタイムアウトした場合，トレースログを出力します。	0
com.cosminexus.admin.auth.trace.rotate	トレースファイル面数を，"1"から"16"までの数字で指定します。	4
com.cosminexus.admin.auth.trace.size	1 トレースファイル当たりの最大サイズを，"4096"から"2147483647"の数字で指定します。 一つのログファイルのサイズがここで指定したサイズを超えた場合，ログは次の面番号の付いたログファイルに記録されます。最後のログファイル（面数の番号が付いたログファイル）のサイズが 1 ファイル当たりの最大サイズに達すると，面の番号 1 のログファイルへ上書きします。	65536
com.cosminexus.admin.auth.sfo.disable	セッションフェイルオーバ用フィルタが設定されていた場合，統合ユーザ管理のセッションフェイルオーバ対応機能を無効にします。 true を指定した場合： セッションフェイルオーバ対応機能を無効にします。 false を指定した場合： セッションフェイルオーバ対応機能を有効にします。	false

14.3　シングルサインオン用認証情報の CSV 形式ファイル

シングルサインオン用の認証情報は，CSV 形式ファイルを使用して作成します。CSV 形式ファイルについて，次に示します。

14.3.1　CSV 形式ファイルの基本仕様

各項目の区切りには半角のコンマ（,）を使用します。また，レコードの区切りは改行です。

各項目は，ダブルクォーテーション（"）で囲んでも囲まなくてもコンマで区切られた文字列を 1 データとして扱います。ただし，項目にコンマを記述する場合は，項目全体をダブルクォーテーションで囲みます。

（例）RDN 名に「ou=APS,o=CorpHitachi」を指定する場合

```
...,”ou=APS,o=Corp”,...
```

また，項目にダブルクォーテーションを記述する場合は，ダブルクォーテーションを 2 文字記述し，さらに項目全体をダブルクォーテーションで囲みます。

（例）Alias に「pass"wd」を指定する場合

```
...,”pass””wd”,...
```

なお，コンマ（,）の前後のスペースは各項目に含めます。

14.3.2　ユーザ情報を取得するための定義ファイル

(1)　CSV 形式ファイルの追加仕様

属性の一覧を指定するファイルのフォーマットでは，「14.3.1　CSV 形式ファイルの基本仕様」で示した仕様に次の仕様が付加されます。

- 項目の指定順序は決められています。
- コンマ（,）を続けて指定した場合は，その位置のオプションが省略されたものとします。

(2)　指定方法

改行までを 1 行として，1 行ごとに次に示す項目をコンマで区切って指定します。

形式	項目		
形式 1	#		
形式 2	属性名	Alias	サブコンテキスト

形式 1

注釈（コメント）を指定します。行の先頭（1 カラム目）が”#”であれば行末までを注釈と見なします。

形式 2

次の表に示す情報を 1 行で指定します。

表 14-2　指定する情報（ユーザ情報を取得するための定義ファイル）

機能	意味	属性
属性名	英字で始まる，英字（ASCII 文字），数字，ハイフンで指定します。英字は，大文字と小文字を区別しません。	必須
Alias	プログラムで参照するための名称を指定します。	任意
サブコンテキスト	認証したユーザのエントリ以外のエントリの情報を取得する場合に，取得するエントリに対するユーザのエントリからの RDN を指定します。	任意

14.3.3　ユーザ情報を追加および変更するための定義ファイル

LDAP ディレクトリサーバのエントリのオブジェクトクラスを指定するファイルです。

(1)　CSV 形式ファイルの追加仕様

属性の一覧を指定するファイルのフォーマットでは，「14.3.1　CSV 形式ファイルの基本仕様」で示した仕様に次の仕様が付加されます。

- 項目の指定順序は決められています。
- コンマ（,）を続けて指定した場合は，その位置のオプションが省略されたものとします。

(2)　指定方法

改行までを 1 行として，1 行ごとに次に示す項目をコンマで区切って指定します。

形式	項目	
形式 1	#	
形式 2	サブコンテキスト	オブジェクトクラス[,オブジェクトクラス…]

形式 1

注釈（コメント）を指定します。行の先頭（1 カラム目）が”#”であれば行末までを注釈と見なします。

形式 2

次の表に示す情報を 1 行で指定します。

表 14-3　指定する情報（ユーザ情報を追加および変更するための定義ファイル）

機能	意味	属性
サブコンテキスト	認証に使用するユーザエントリからの RDN を指定します。省略した場合は，ユーザエントリを仮定します。	任意
オブジェクトクラス	サブコンテキストのオブジェクトクラスを指定します。コンマで区切って複数指定できます。	必須

14.3.4　ユーザマッピングと認証情報の定義ファイル

(1)　CSV 形式ファイルの追加仕様

属性の一覧を指定するファイルのフォーマットでは，「14.3.1　CSV 形式ファイルの基本仕様」で示した仕様に次の仕様が付加されます。

- 1 行目にヘッダ情報，2 行目以降に登録するデータの行となります。
- 各項目の内容はヘッダによって決まります。
- コンマ（,）を続けて指定した場合は，その位置のオプションが省略されたものとします。

(2)　指定方法

1 行目に指定する内容

次の表に示すヘッダ情報を指定します。各項目は ASCII 文字で入力し，半角のコンマ（,）で区切ります。項目 ID の指定順序は任意です。

表 14-4　指定するヘッダ情報（ユーザマッピングと認証情報の定義ファイル）

項目 ID	指定項目	指定内容	属性
REALMNAME	登録者の識別子	レルム名を指定します。ここで指定された名前の下にユーザエントリが作成されます。	必須
USERID		ユーザ ID	必須
SECRETDATA	認証情報	暗号化して保存するデータ	任意
PUBLICDATA		暗号化する必要のない保存データ	任意
LINK_xxxx	接続先システムのユーザ	ユーザ管理を持つアプリケーションのユーザ名を指定します（xxxx は各 REALMNAME を指定します）。	任意
OPERATION	行に対するコマンド	ラインオペレーションを指定します。この指定は，一つのファイル内で，追加，変更，および削除を共用して指定できます。	任意

項目 ID 以外の名称が指定された場合は，その欄は無視されます。

LINK_xxxx はシングルサインオン情報リポジトリに登録されているレルムごとに作成される項目 ID です。

2 行目以降

実際に登録するデータを，半角のコンマ（,）で区切って指定します。

(3) JAAS 対応ユーザ管理を持つアプリケーションのユーザ定義

LINK_xxxx 項目 ID の xxxx 部分に接続先のユーザ管理を持つアプリケーションを表すレルム名が表示されます。この項目欄にユーザ ID を指定して接続先を定義します。ここに指定する内容は，次の操作で追加，変更できます。

追加

対象の LINK_xxxx 項目に接続先のユーザ管理を持つアプリケーション（レルム）のユーザ ID を記述します。

変更

対象の LINK_xxxx 項目に接続先のユーザ管理を持つアプリケーション（レルム）のユーザ ID に変更します。

解除

対象の LINK_xxxx 項目からユーザ ID を削除（何も記述しない）します。

14.3.5 CSV 形式ファイルの記述例

例えば，レルム名が「Portal」のユーザ「taro」「hanako」「jirou」は，レルム名「RealmA」の「k010000」ユーザを使用します。レルム名「J2EE」のユーザ「hanako」は「Admin」ユーザを，ユーザ「jirou」は「DBMgr」ユーザを使用するとします。この場合，CSV 形式ファイルは次のようになります（この例では，ユーザ ID とパスワードは同じものであると仮定しています）。

REALMNAME	USERID	SECRETDATA	PUBLICDATA	LINK_J2EE	LINK_REALMA
Portal	taro	taro	developer		k010000
Portal	hanako	hanako		Admin	k010000
Portal	jirou	jirou		DBMgr	k010000
RealmA	k010000	k010000			
J2EE	Admin	Admin			
J2EE	DBMgr	DBMgr			

記述例

```
REALMNAME,USERID,SECRETDATA,PUBLICDATA,LINK_J2EE,LINK_REALMA
Portal,taro,taro,developer,,k010000
Portal,hanako,hanako,,Admin,k010000
Portal,jirou,jirou,,DBMgr,k010000
RealmA,k010000,k010000,,,
```

```
J2EE,Admin,Admin,,,
J2EE,DBMgr,DBMgr,,,
```

14.3.6 ラインオペレーション

ラインオペレーションとは CSV 形式ファイルの項目”OPERATION”の項目に指定された値に従ってシングルサインオン情報リポジトリに行を登録，変更，および削除する機能です。項目”OPERATION”は管理者が任意に挿入できますが，有効にしたい場合は必ず ssoimport コマンドに-x オプションを指定しなければなりません。また，-x オプションを-a,-m,-d オプションと併用することはできません。

項目”OPERATION”に指定できるオペレーションと用途を次の表に示します。

表 14–5 項目”OPERATION”に指定できるオペレーションと用途

オペレーション	用途
A または a	指定した行の内容をシングルサインオン情報リポジトリに追加します。もし，シングルサインオン情報リポジトリ内にユーザエントリがある場合は，警告メッセージを出力して処理を継続します。
M または m	指定した行の内容にシングルサインオン情報リポジトリを上書きします。もし，シングルサインオン情報リポジトリ内にユーザエントリがない場合は，ユーザエントリを追加します。
D または d	指定した行の内容をシングルサインオン情報リポジトリから削除します。もし，シングルサインオン情報リポジトリ内にユーザエントリがない場合は，警告メッセージを出力して処理を継続します。

注

オペレーションは大文字と小文字の制限はありません。また，オペレーション前後の空文字は無視されますが，タブは使用できません。

オペレーション欄に，A，a，M，m，D，および d 以外が指定された場合は，Warning が発生し該当する行はスキップされます。このような場合は，適切なオペレーションが指定されているかを確認し再度登録処理をする必要があります。

項目”OPERATION”を追加した場合の CSV 形式ファイルを次に示します。

OPERATION	REALMNAME	USERID	SECRETDATA	PUBLICDATA	LINK_J2EE
A	Portal	taro	taro	developer	
D	Portal	hanako	hanako		Admin
M	Portal	jirou	jirou		DBMgr
	RealmA	k010000	k010000		
M	J2EE	Admin	Admin		

記述例

```
OPERATION,REALMNAME,USERID,SECRETDATA,PUBLICDATA,LINK_J2EE
A,Portal,taro,taro,developer,
D,Portal,hanako,hanako,,Admin
M,Portal,jirou,jirou,,DBMgr
```

```
,RealmA,k010000,k010000,,
M,J2EE,Admin,Admin,,
```

15 統合ユーザ管理フレームワークで使用する API

この章では，統合ユーザ管理フレームワークで使用する API および例外クラスについて説明します。

15.1　統合ユーザ管理フレームワークで使用する API の一覧

統合ユーザ管理フレームワークのライブラリを利用してユーザ認証を実装する場合に使用する API および例外クラスの一覧を次の表に示します。

表 15–1　統合ユーザ管理フレームワークで使用する API および例外クラスの一覧

クラス・インタフェース名	機能	API の種別
AttributeEntry クラス	属性名と Alias を対で管理します。	ユーザ認証ライブラリ
ChangeDataFailedException クラス	SSODataListener インタフェースの実装クラスが呼び出す例外クラスです。	シングルサインオンライブラリ（例外クラス）
DelegationLoginModule クラス	JAAS のログインモジュールの実装クラスです。カスタムログインモジュールを呼び出します。	標準ログインモジュール
LdapSSODataManager クラス	LDAP ディレクトリサーバのシングルサインオン情報リポジトリの情報を参照または更新します。	シングルサインオンライブラリ
LdapUserDataManager クラス	LDAP ディレクトリサーバの，ユーザ情報リポジトリの情報を参照または更新します。	ユーザ認証ライブラリ
LdapUserEnumeration インタフェース	ユーザ ID の一覧を参照します。	ユーザ認証ライブラリ
LoginUtil クラス	統合ユーザ管理のセッション内でログインしているユーザの有無を調べます。	ユーザ認証ライブラリ
ObjectClassEntry クラス	LDAP ディレクトリサーバのエントリのオブジェクトクラスを格納します。	ユーザ認証ライブラリ
PasswordCryptography インタフェース	ユーザが入力したパスワードを暗号化します。	ユーザ認証ライブラリ
PasswordUtil クラス	ユーザが入力したパスワードを変更します。	ユーザ認証ライブラリ
Principal インタフェース	WebPasswordLoginModule が認証したときのユーザ ID を参照します。	ユーザ認証ライブラリ
SSOData クラス	シングルサインオン用認証情報を格納します。	シングルサインオンライブラリ
SSODataEvent クラス	シングルサインオン用認証情報の更新内容を格納します。	シングルサインオンライブラリ
SSODataListener インタフェース	シングルサインオン用認証情報の更新を通知します。	シングルサインオンライブラリ
SSODataListenerException クラス	シングルサインオン用認証情報リスナクラスで例外が発生した場合に呼び出される例外クラスです。	シングルサインオンライブラリ（例外クラス）
UserAttributes インタフェース	WebPasswordLoginModule が認証したときに作成した Credential を参照します。	ユーザ認証ライブラリ
UserData クラス	ユーザ情報を格納します。	ユーザ認証ライブラリ
WebCertificateCallback クラス	JAAS の Callback の実装クラスです。Web サーバの SSL 認証した結果の情報を格納します。	ユーザ認証ライブラリ

クラス・インタフェース名	機能	API の種別
WebCertificateHandler クラス	JAAS の CallbackHandler の実装クラスです。Web サーバの SSL 認証した結果で必要な情報を読み込みます。	ユーザ認証ライブラリ
WebCertificateLoginModule クラス	JAAS のログインモジュールの実装クラスです。Web サーバで認証された証明書からユーザ属性を求めます。	標準ログインモジュール
WebLogoutCallback クラス	JAAS の Callback の実装クラスです。ログアウトするユーザを格納します。	ユーザ認証ライブラリ
WebLogoutHandler クラス	JAAS の CallbackHandler の実装クラスです。ログアウトに必要なユーザを読み込みます。	ユーザ認証ライブラリ
WebPasswordCallback クラス	JAAS の Callback の実装クラスです。パスワードなどの認証情報を格納します。	ユーザ認証ライブラリ
WebPasswordHandler クラス	JAAS の CallbackHandler の実装クラスです。パスワード認証に必要な情報を読み込みます。	ユーザ認証ライブラリ
WebPasswordJDBCLoginModule クラス	JAAS のログインモジュールの実装クラスです。JDBC を使ってデータベースにアクセスし，パスワード認証をします。	標準ログインモジュール
WebPasswordLDAPLoginModule クラス	JAAS のログインモジュールの実装クラスです。LDAP ディレクトリサーバにバインドした結果で認証をします。	標準ログインモジュール
WebPasswordLoginModule クラス	JAAS のログインモジュールの実装クラスです。Web アプリケーションのためのパスワード認証をします。	標準ログインモジュール
WebSSOCallback クラス	シングルサインオンライブラリが提供する JAAS の Callback の実装クラスです。WebSSOLoginModule で必要な情報を知るために使用します。	シングルサインオンライブラリ
WebSSOHandler クラス	シングルサインオンライブラリが提供する JAAS の CallbackHandler の実装クラスです。WebSSOLoginModule で必要な情報を読み込みます。	シングルサインオンライブラリ
WebSSOLoginModule クラス	JAAS のログインモジュールの実装クラスです。シングルサインオンをするためにほかのログインモジュールを呼び出します。	標準ログインモジュール
例外クラス	統合ユーザ管理で使用する API の例外クラスです。	例外クラス

15.2 AttributeEntry クラス

説明

ユーザ管理リポジトリから取得する属性の属性名，別名（Alias），およびユーザ管理コンテキストからのサブコンテキストのタプルを表すクラスです。ユーザ認証後，指定した属性は Subject の Public Credential に別名で関連づけられます。別名を指定しなかった場合は，属性名で関連づけられます。

AttributeEntry クラスのパッケージ名は，com.cosminexus.admin.auth です。

形式

```
class AttributeEntry
{
public AttributeEntry(String attr,
                      String alias,
                      String subcontext);
public AttributeEntry(String attr,
                      String alias);
public AttributeEntry(String attr);
public AttributeEntry();

public String getAlias();
public String getAttributeName();
public String getSubcontext();
public void setAlias(String alias);
public void setAttributeName(String attr);
public void setSubcontext(String subcontext);
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
AttributeEntry コンストラクタ	AttributeEntry クラスのインスタンスを生成します。
getAlias メソッド	setAlias メソッドまたはコンストラクタで指定した別名を取得します。
getAttributeName メソッド	setAttributeName メソッドまたはコンストラクタで指定した属性名を取得します。
getSubcontext メソッド	setSubcontext メソッドまたはコンストラクタで指定したサブコンテキストを取得します。
setAlias メソッド	パラメタに指定された別名をオブジェクトに保管します。
setAttributeName メソッド	パラメタに指定された属性名をオブジェクトに保管します。
setSubcontext メソッド	パラメタに指定されたサブコンテキストをオブジェクトに保管します。

AttributeEntry コンストラクタ

説明

AttributeEntry クラスのインスタンスを作るためのコンストラクタです。

形式

```
public AttributeEntry(String attr,
                      String alias,
                      String subcontext);

public AttributeEntry(String attr,
                      String alias);

public AttributeEntry(String attr);

public AttributeEntry();
```

パラメタ

attr：

　リポジトリに格納されている属性名を指定します。

alias：

　属性名に対応する別名（Alias）を指定します。

subcontext：

　サブコンテキストを指定します。

例外

なし

getAlias メソッド

説明

setAlias メソッドまたはコンストラクタで指定した内容を取得します。値を保管していないときに getAlias メソッドを呼び出すと null が返却されます。

形式

```
public String getAlias();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getAttributeName メソッド

説明

setAttributeName メソッドまたはコンストラクタで指定した内容を取得します。値を保管していないときに getAttributeName メソッドを呼び出すと null が返却されます。

形式

```
public String getAttributeName();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getSubcontext メソッド

説明

setSubcontext メソッドまたはコンストラクタで指定した内容を取得します。値を保管していないときに getSubcontext メソッドを呼び出すと null が返却されます。

形式

```
public String getSubcontext();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

setAlias メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合にsetAlias メソッドを呼び出したときは上書きされます。

形式

```
public void setAlias(String alias);
```

パラメタ

alias：
　属性名に対応する別名（Alias）を指定します。

例外

なし

戻り値

なし

setAttributeName メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合にsetAttributeName メソッドを呼び出したときは上書きされます。

形式

```
public void setAttributeName(String attr);
```

パラメタ

attr：

属性名を指定します。

例外

なし

戻り値

なし

setSubcontext メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合に setSubcontext メソッドを呼び出したときは上書きされます。

形式

```
public void setSubcontext(String subcontext);
```

パラメタ

subcontext：

サブコンテキストを指定します。

例外

なし

戻り値

なし

15.3 ChangeDataFailedException クラス

説明

SSODataListener インタフェースの実装クラスが，データの追加，修正，または削除に失敗したときに呼び出す例外クラスです。

ChangeDataFailedException クラスのパッケージ名は，com.cosminexus.admin.auth.api.repository.event です。

形式

```
class ChangeDataFailedException extends UAException
{
public ChangeDataFailedException();
public ChangeDataFailedException(String msg);
}
```

コンストラクタ一覧

コンストラクタ名	機能
ChangeDataFailedException コンストラクタ	ChangeDataFailedException クラスのインスタンスを生成します。

ChangeDataFailedException コンストラクタ

説明

パラメタに指定されたエラーメッセージを使用してインスタンスを生成します。

形式

```
public ChangeDataFailedException();

public ChangeDataFailedException(String msg);
```

パラメタ

msg：

エラーメッセージを指定します。

例外

なし

15.4 DelegationLoginModule クラス

説明

ユーザ認証ライブラリが提供する JAAS のログインモジュールの実装クラスです。カスタムログインモジュールを呼び出します。

パッケージ名は com.cosminexus.admin.auth.login です。

15.5 LdapSSODataManager クラス

説明

LDAP ディレクトリサーバのシングルサインオン情報リポジトリに格納されている情報を参照または更新するクラスです。

LdapSSODataManager クラスのパッケージ名は，
com.cosminexus.admin.auth.api.repository.ldap です。

形式

```
class LdapSSODataManager
{
  public LdapSSODataManager(String realm);

  public LdapUserEnumeration listUsers()
    throws NamingException;
  public LdapUserEnumeration listUsers(String uid)
    throws NamingException;
  public SSOData getSSOData(String uid)
    throws NamingException;
  public void addSSOData(String uid,
                         SSOData ssoData)
    throws SSODataListenerException, NamingException,
    CryptoException, UnsatisfiedLinkError, SecurityException;
  public void removeSSOData(String uid)
    throws SSODataListenerException, NamingException,
    CryptoException, UnsatisfiedLinkError, SecurityException;
  public void modifySSOData(String uid,
                            SSOData ssoData)
    throws SSODataListenerException, NamingException,
    CryptoException, UnsatisfiedLinkError, SecurityException;
  public SSODataListener[] getSSODataListeners();
  public void addSSODataListener(SSODataListener listener);
  public void removeSSODataListener(SSODataListener listener);
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
LdapSSODataManager コンストラクタ	LdapSSODataManager クラスのインスタンスを生成します。
addSSOData メソッド	シングルサインオン用認証情報を追加します。
addSSODataListener メソッド	シングルサインオン用認証情報リスナを登録します。
getSSOData メソッド	シングルサインオン用認証情報を取得します。
getSSODataListeners メソッド	SSODataListener オブジェクトの配列を取得します。
listUsers メソッド（形式 1）	すべてのユーザ ID の一覧を取得します。
listUsers メソッド（形式 2）	ユーザ ID の一覧を取得します。
modifySSOData メソッド	シングルサインオン用認証情報を修正します。

コンストラクタ・メソッド名	機能
removeSSOData メソッド	シングルサインオン用認証情報を削除します。
removeSSODataListener メソッド	SSODataListener オブジェクトを削除します。

LdapSSODataManager コンストラクタ

説明

インスタンスを生成します。

形式

```
public LdapSSODataManager(String realm);
```

パラメタ

realm：

生成したインスタンスがアクセス対象にするレルム名を指定します。

例外

なし

addSSOData メソッド

説明

指定したユーザのシングルサインオン用認証情報を追加します。指定したユーザのシングルサインオン用認証情報がすでにある場合は，例外が発生します。

このオブジェクトに登録されているすべてのシングルサインオン用認証情報リスナの ssoDataAdded メソッドが呼び出されます。

形式

```
public void addSSOData(String uid,
                       SSOData ssoData)
  throws SSODataListenerException, NamingException,
  CryptoException, UnsatisfiedLinkError, SecurityException;
```

パラメタ

uid：

ユーザ ID を指定します。

ssoData：

シングルサインオン用認証情報を格納した SSOData オブジェクトを指定します。

例外

com.cosminexus.admin.auth.api.repository.event.SSODataListenerException：

他システムの認証情報更新に失敗しました。

com.cosminexus.admin.auth.CryptoException：

暗号鍵ファイルの読み込みに失敗しました。または，誤った暗号鍵ファイルを使用したため SecretData の復号化に失敗しました。

java.lang.UnsatisfiedLinkError：

シングルサインオンライブラリの読み込みに失敗しました。

java.lang.SecurityException：

SecurityManager が存在し，SecurityManager の checkRead メソッドでファイルへの読み込みアクセスが拒否されました。

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameAlreadyBoundException：

指定したユーザのシングルサインオン用認証情報がすでにあります。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

なし

addSSODataListener メソッド

説明

シングルサインオン用認証情報を追加，修正，または削除したとき，他システムに変更を通知するためのシングルサインオン用認証情報リスナをこのオブジェクトに登録します。

形式

```
public void addSSODataListener(SSODataListener listener);
```

パラメタ

listener：

SSODataListener オブジェクトを指定します。null を指定した場合は何もしません。

例外

なし

戻り値

なし

getSSOData メソッド

説明

シングルサインオン用認証情報を取得します。

形式

```
public SSOData getSSOData(String uid)
  throws NamingException;
```

パラメタ

uid：

ユーザ ID を指定します。

例外

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameNotFoundException：

指定したユーザ ID がありません。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

シングルサインオン用認証情報を格納した SSOData オブジェクトを返却します。

getSSODataListeners メソッド

説明

このオブジェクトに登録されている SSODataListener オブジェクトの配列を取得します。登録されていない場合は大きさ 0 の配列を返却します。

形式

```
public SSODataListener[] getSSODataListeners();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトに登録されている SSODataListener オブジェクトの配列を返却します。

listUsers メソッド（形式 1）

説明

すべてのユーザ ID の一覧を取得します。addSSOData メソッドまたは removeSSOData メソッドを呼び出した場合，以前に返却された LdapUserEnumeration オブジェクトにその結果が反映されるかどうかは不定です。

形式

```
public LdapUserEnumeration listUsers()
  throws NamingException;
```

パラメタ

なし

例外

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

ユーザ ID の一覧を格納した LdapUserEnumeration オブジェクトを返却します。

listUsers メソッド（形式 2）

説明

ユーザ ID の一覧を取得します。addSSOData メソッドまたは removeSSOData メソッドを呼び出した場合，以前に返却された LdapUserEnumeration オブジェクトにその結果が反映されるかどうかは不定です。

形式

```
public LdapUserEnumeration listUsers(String uid)
  throws NamingException;
```

パラメタ

uid：

ユーザ ID を指定します。ユーザ ID にはワイルドカード（*）を含めることができます。このパラメタを省略した場合や null を指定した場合は，すべてのユーザ ID の一覧を取得します。

例外

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

ユーザ ID の一覧を格納した LdapUserEnumeration オブジェクトを返却します。

modifySSOData メソッド

説明

シングルサインオン用認証情報を修正します。指定したユーザがない場合は例外が発生します。

このオブジェクトに登録されているすべてのシングルサインオン用認証情報リスナの ssoDataModified メソッドが呼び出されます。

このメソッドでは，SSOData オブジェクトの生成後，変更された認証情報だけが既存のものに上書きされます。

例えば，リポジトリの既存のシングルサインオン用認証情報が次に示す要素を持っていたとします。

認証情報名	SecretData	PublicData	マッピング	
			レルム	ユーザ ID
値	secret	public	RealmA	user1
			RealmB	admin

このとき，次のコードで生成した SSOData オブジェクトをこのメソッドのパラメタに指定します。

```
SSOData data = new SSOData();
data.setMapping("RealmA", "user2");
```

すると，リポジトリのシングルサインオン用認証情報は次のように変更されます。

認証情報名	SecretData	PublicData	マッピング	
			レルム	ユーザ ID
値	secret	public	RealmA	user2
	－	－	－	－

（凡例）－：情報がないことを示します。

形式

```
public void modifySSOData(String uid,
                          SSOData ssoData)
  throws SSODataListenerException, NamingException,
  CryptoException, UnsatisfiedLinkError, SecurityException;
```

パラメタ

uid：

ユーザ ID を指定します。

ssoData：

シングルサインオン用認証情報を格納した SSOData オブジェクトを指定します。

例外

com.cosminexus.admin.auth.api.repository.event.SSODataListenerException：

他システムの認証情報更新に失敗しました。

com.cosminexus.admin.auth.CryptoException：

暗号鍵ファイルの読み込みに失敗しました。または，誤った暗号鍵ファイルを使用したため SecretData の復号化に失敗しました。

java.lang.UnsatisfiedLinkError：

シングルサインオンライブラリの読み込みに失敗しました。

java.lang.SecurityException：

SecurityManager が存在し，SecurityManager の checkRead メソッドでファイルへの読み込みアクセスが拒否されました。

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameNotFoundException：

指定したユーザ ID がありません。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

なし

removeSSOData メソッド

説明

シングルサインオン用認証情報を削除します。指定したユーザがない場合は例外が発生します。

このオブジェクトに登録されているすべてのシングルサインオン用認証情報リスナの ssoDataRemoved メソッドが呼び出されます。

形式

```
public void removeSSOData(String uid)
  throws SSODataListenerException, NamingException,
  CryptoException, UnsatisfiedLinkError, SecurityException;
```

パラメタ

uid：

ユーザ ID を指定します。

例外

com.cosminexus.admin.auth.api.repository.event.SSODataListenerException：

他システムの認証情報更新に失敗しました。

com.cosminexus.admin.auth.CryptoException：

暗号鍵ファイルの読み込みに失敗しました。または誤った暗号鍵ファイルを使用したため SecretData の復号化に失敗しました。

java.lang.UnsatisfiedLinkError：

シングルサインオンライブラリの読み込みに失敗しました。

java.lang.SecurityException：

SecurityManager が存在し，SecurityManager の checkRead メソッドでファイルへの読み込みアクセスが拒否されました。

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameNotFoundException：

指定したユーザ ID がありません。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

なし

removeSSODataListener メソッド

説明

指定した SSODataListener オブジェクトをこのオブジェクトから削除します。指定したオブジェクトが登録されていない場合は何もしません。

形式

```
public void removeSSODataListener(SSODataListener listener);
```

パラメタ

listener：

SSODataListener オブジェクトを指定します。

例外

なし

戻り値

なし

15.6 LdapUserDataManager クラス

説明

LDAP ディレクトリサーバのユーザ情報リポジトリに格納されている情報を参照または更新するクラスです。

このクラスのオブジェクトごとに，addUserData メソッド，modifyUserData メソッド，removeUserData メソッド，および getUserData メソッドで排他制御をします。異なるオブジェクトで同時に同じリポジトリを操作しないでください。

LdapUserDataManager クラスのパッケージ名は，com.cosminexus.admin.auth.api.repository.ldap です。

形式

```
class LdapUserDataManager
{
  public LdapUserDataManager(String name)
    throws ConfigError;
  public LdapUserDataManager(String name,
                             AttributeEntry[] aliases)
    throws ConfigError, FormatError;
  public LdapUserDataManager(String name,
                             String aliasesFile)
    throws ConfigError, FormatError, IOException,
           FileNotFoundException,
    SecurityException;
  public LdapUserDataManager(String name,
                             AttributeEntry[] aliases,
                             ObjectClassEntry[] ocEntries)
    throws ConfigError, FormatError;
  public LdapUserDataManager(String name,
                             AttributeEntry[] aliases,
                             String objclassesFile)
    throws ConfigError, FormatError, IOException,
           FileNotFoundException,SecurityException;
  public LdapUserDataManager(String name,
                             String aliasesFile,
                             ObjectClassEntry[] ocEntries)
    throws ConfigError, FormatError, IOException,
           FileNotFoundException,SecurityException;
  public LdapUserDataManager(String name,
                             String aliasesFile,
                             String objclassesFile)
    throws ConfigError, FormatError, IOException,
           FileNotFoundException,SecurityException;

  public LdapUserEnumeration listUsers()
    throws NamingException;
  public LdapUserEnumeration listUsers(String uid)
    throws NamingException;
  public UserData getUserData(String uid)
    throws NamingException;
  public void addUserData(String uid,
                          UserData userData)
    throws ObjectClassError, NamingException;
```

```
  public void addUserData(String uid,
                          UserData userData,
                          String name, String value)
    throws ObjectClassError, NamingException;
  public void removeUserData(String uid)
    throws NamingException;
  public void modifyUserData(String uid, UserData userData)
    throws ObjectClassError, NamingException;
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
LdapUserDataManager コンストラクタ	LdapUserDataManager クラスのインスタンスを生成します。
addUserData メソッド（形式 1）	ユーザを追加します。ユーザエントリの DN に uid を使用します。
addUserData メソッド（形式 2）	ユーザを追加します。ユーザエントリの DN に任意の属性を使用します。
getUserData メソッド	ユーザ情報を取得します。
listUsers メソッド（形式 1）	すべてのユーザ ID の一覧を取得します。
listUsers メソッド（形式 2）	ユーザ ID の一覧を取得します。
modifyUserData メソッド	ユーザ情報を修正します。
removeUserData メソッド	ユーザを削除します。

LdapUserDataManager コンストラクタ

説明

LdapUserDataManager クラスのインスタンスを生成します。ユーザ属性情報やオブジェクトクラスは，オブジェクトやファイルで指定できます。また，省略もできます。

形式

```
public LdapUserDataManager(String name)
  throws ConfigError;

public LdapUserDataManager(String name,
                           AttributeEntry[] aliases)
  throws ConfigError, FormatError;

public LdapUserDataManager(String name,
                           String aliasesFile)
  throws ConfigError, FormatError, IOException, FileNotFoundException,
  SecurityException;

public LdapUserDataManager(String name,
                           AttributeEntry[] aliases,
                           ObjectClassEntry[] ocEntries)
```

```
  throws ConfigError, FormatError;

public LdapUserDataManager(String name,
                           AttributeEntry[] aliases,
                           String objclassesFile)
  throws ConfigError, FormatError, IOException, FileNotFoundException,
  SecurityException;

public LdapUserDataManager(String name,
                           String aliasesFile,
                           ObjectClassEntry[] ocEntries)
  throws ConfigError, FormatError, IOException, FileNotFoundException,
  SecurityException;

public LdapUserDataManager(String name,
                           String aliasesFile,
                           String objclassesFile)
  throws ConfigError, FormatError, IOException, FileNotFoundException,
  SecurityException;
```

パラメタ

name：

アクセス対象にする LDAP ディレクトリサーバの設定名を指定します。設定名はユーザ管理のコンフィグレーションファイルで定義します。

aliases：

参照または更新するユーザ属性情報として，AttributeEntry オブジェクトの配列を指定します。指定した内容で必要な情報がない場合は FormatError 例外が発生します。このパラメタを省略した場合や null を指定した場合，属性を参照および更新できません。ただし，パスワードは更新できます。

aliasesFile：

参照または更新するユーザ属性情報として，ファイル名を指定します。指定した内容で必要な情報がない場合は FormatError 例外が発生します。このパラメタを省略した場合や null を指定した場合，属性を参照および更新できません。ただし，パスワードは更新できます。

ocEntries：

LDAP ディレクトリサーバにエントリを作成したり，修正したりするときに使用するオブジェクトクラスの配列を指定します。指定した内容で必要な情報がない場合は FormatError 例外が発生します。このパラメタを省略した場合や null を指定した場合，ユーザ情報の追加や変更時に ObjectClassError 例外が発生します。

objclassesFile：

LDAP ディレクトリサーバのエントリのオブジェクトクラスが定義されたファイル名を指定します。指定した内容で必要な情報がない場合は FormatError 例外が発生します。このパラメタを省略した場合や null を指定した場合，ユーザ情報の追加や変更時に ObjectClassError 例外が発生します。

例外

java.io.FileNotFoundException：

ファイルがない，ファイルではなくディレクトリである，またはそれ以外の何かの理由で開くことができません（FileInputStream クラスのコンストラクタで例外が呼び出された場合）。

java.lang.SecurityException：

SecurityManager が存在し，SecurityManager の checkRead メソッドでファイルへの読み込みアクセスが拒否されました。

java.io.IOException：

ファイルの読み込みに失敗しました。

com.cosminexus.admin.common.ConfigError：

設定名が統合ユーザ管理のコンフィグレーションファイルにありません。

com.cosminexus.admin.common.FormatError：

aliases，aliasesFile，ocEntries および objclassesFile に指定された内容で必要な情報がありません。または余分に指定されています。

addUserData メソッド（形式 1）

説明

ユーザを追加します。すでにユーザがいる場合は例外が発生します。

LDAP ディレクトリサーバに作成するユーザエントリの DN には，ユーザ ID の属性（uid）と値が使用されます。

ユーザエントリはベース DN の直下に作成されます。また，コンストラクタで指定したユーザ属性情報にサブコンテキストの属性が含まれる場合，サブコンテキストのエントリも作成されます。

このメソッドを呼び出した結果，サブコンテキストの更新中に例外が発生した場合，ユーザ情報は不完全に更新された状態になっています。その場合，原因を取り除いてから removeUserData メソッドで該当ユーザを削除し，再びこのメソッドを呼び出してください。

形式

```
public void addUserData(String uid,
                        UserData userData)
  throws ObjectClassError, NamingException;
```

パラメタ

uid：

ユーザ ID を指定します。

userData：

ユーザ情報を格納した UserData オブジェクトを指定します。

例外

com.cosminexus.admin.auth.api.repository.ldap.ObjectClassError：

LDAP ディレクトリサーバにエントリを作成するために必要なオブジェクトクラスが指定されていません。

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameAlreadyBoundException：

指定したユーザ ID がすでにあります。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

なし

注意事項

getUserData メソッドで取得した UserData オブジェクトにパスワードは格納されていません。そのため，getUserData メソッドで取得した UserData オブジェクトをそのまま addUserData メソッドのパラメタに指定しても，ユーザの完全なコピーをすることはできません。パスワードを新たに設定してください。

addUserData メソッド（形式 2）

説明

ユーザを追加します。すでにユーザがいる場合は例外が発生します。

LDAP ディレクトリサーバに作成するユーザエントリの DN には，このメソッドで指定した属性名と値が使用されます。

ユーザエントリはベース DN の直下に作成されます。また，コンストラクタで指定したユーザ属性情報にサブコンテキストの属性が含まれる場合，サブコンテキストのエントリも作成されます。

このメソッドを呼び出した結果，サブコンテキストの更新中に例外が発生した場合，ユーザ情報は不完全に更新された状態になっています。その場合，原因を取り除いてから removeUserData メソッドで該当ユーザを削除し，再びこのメソッドを呼び出してください。

形式

```
public void addUserData(String uid,
                        UserData userData,
                        String name,
                        String value)
  throws ObjectClassError, NamingException;
```

パラメタ

uid：

ユーザ ID を指定します。

userData：

ユーザ情報を格納した UserData オブジェクトを指定します。

name：

ユーザエントリの DN に使用する属性名を指定します。

value：

ユーザエントリの DN に使用する属性値を指定します。

例外

com.cosminexus.admin.auth.api.repository.ldap.ObjectClassError：

LDAP ディレクトリサーバにエントリを作成するために必要なオブジェクトクラスが指定されていません。

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameAlreadyBoundException：

指定したユーザ ID がすでにあります。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

なし

注意事項

getUserData メソッドで取得した UserData オブジェクトにパスワードは格納されていません。そのため，getUserData メソッドで取得した UserData オブジェクトをそのまま addUserData メソッドのパラメタに指定しても，ユーザの完全なコピーをすることはできません。パスワードを新たに設定してください。

getUserData メソッド

説明

ユーザ情報を取得します。取得した UserData オブジェクトにパスワードは格納されていません。

形式

```
public UserData getUserData(String uid)
  throws NamingException;
```

パラメタ

uid：

ユーザ ID を指定します。

例外

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameNotFoundException：

指定したユーザ ID がありません。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

ユーザ情報を格納した UserData オブジェクトを返却します。

listUsers メソッド（形式 1）

説明

すべてのユーザ ID の一覧を取得します。addUserData メソッドまたは removeUserData メソッドを呼び出した場合，以前に返却された LdapUserEnumeration オブジェクトにその結果が反映されるかどうかは不定です。

形式

```
public LdapUserEnumeration listUsers()
  throws NamingException;
```

パラメタ

なし

例外

javax.naming.CommunicationException：

　LDAP ディレクトリサーバへの接続に失敗しました。

その他 JNDI の例外：

　バインド DN の指定ミスなどです。

戻り値

ユーザ ID の一覧を格納した LdapUserEnumeration オブジェクトを返却します。

listUsers メソッド（形式 2）

説明

ユーザ ID の一覧を取得します。addUserData メソッドまたは removeUserData メソッドを呼び出した場合，以前に返却された LdapUserEnumeration オブジェクトにその結果が反映されるかどうかは不定です。

形式

```
public LdapUserEnumeration listUsers(String uid)
  throws NamingException;
```

パラメタ

uid：

ユーザ ID を指定します。ユーザ ID にはワイルドカード（*）を含めることができます。このパラメタを省略した場合や null を指定した場合は，すべてのユーザ ID の一覧を取得します。

例外

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

ユーザ ID の一覧を格納した LdapUserEnumeration オブジェクトを返却します。

modifyUserData メソッド

説明

ユーザ情報を修正します。指定したユーザがない場合は例外が発生します。

このメソッドでは，UserData オブジェクトの生成後，変更された属性だけが既存の属性に上書きされます。

例えば，リポジトリの既存のユーザ情報が次に示す属性を持っていたとします。

属性名	full name	tel
値	Sato Taro	111-1111
		222-2222

このとき，次のコードで生成した UserData オブジェクトをこのメソッドのパラメタに指定します。

```
UserData data = new UserData();
data.addAttribute("tel", "111-2222");
```

すると，リポジトリのユーザ情報は次のように変更されます。

属性名	full name	tel
値	Sato Taro	111-2222
		－

（凡例）－：情報がないことを示します。

このメソッドを呼び出した結果，サブコンテキストの更新中に例外が発生した場合，ユーザ情報は不完全に更新された状態になっています。その場合，原因を取り除いてから再びこのメソッドを呼び出してください。

形式

```
public void modifyUserData(String uid,
                           UserData userData)
  throws ObjectClassError, NamingException;
```

パラメタ

uid：

ユーザ ID を指定します。

userData：

ユーザ情報を格納した UserData オブジェクトを指定します。

例外

com.cosminexus.admin.auth.api.repository.ldap.ObjectClassError：

LDAP ディレクトリサーバにエントリを作成するために必要なオブジェクトクラスが指定されていません。

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameNotFoundException：

指定したユーザ ID がありません。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

なし

removeUserData メソッド

説明

ユーザを削除します。指定したユーザがない場合は例外が発生します。LDAP ディレクトリサーバの，指定したユーザエントリ以下のすべてのエントリが削除されます。

このメソッドを呼び出した結果，サブコンテキストの更新中に例外が発生した場合，ユーザ情報は不完全に更新された状態になっています。その場合，原因を取り除いてから再びこのメソッドを呼び出してください。

形式

```
public void removeUserData(String uid)
  throws NamingException;
```

パラメタ

uid：

ユーザ ID を指定します。

例外

javax.naming.CommunicationException：

LDAP ディレクトリサーバへの接続に失敗しました。

javax.naming.NameNotFoundException：

指定したユーザ ID がありません。

その他 JNDI の例外：

バインド DN の指定ミスなどです。

戻り値

なし

15.7 LdapUserEnumeration インタフェース

説明

ユーザ ID の一覧を参照するためのインタフェースです。

LdapUserEnumeration インタフェースのパッケージ名は，
com.cosminexus.admin.auth.api.repository.ldap です。

形式

```
interface LdapUserEnumeration extends java.util.Enumeration
{
  public boolean hasMore()
    throws NamingException;
  public boolean hasMoreElements();
  public String next()
    throws NamingException;
  public Object nextElement();
  public close()
    throws NamingException;
}
```

メソッド一覧

メソッド名	機能
close メソッド	オブジェクトをクローズします。
hasMore メソッド	一覧に次のユーザ ID があるかどうかを判定します。 （NamingException 例外の呼び出し：あり）
hasMoreElements メソッド	一覧に次のユーザ ID があるかどうかを判定します。 （NamingException 例外の呼び出し：なし）
next メソッド	一覧の次のユーザ ID を取得します。 （NamingException 例外の呼び出し：あり 戻り値の型：String 型）
nextElement メソッド	一覧の次のユーザ ID を取得します。 （NamingException 例外の呼び出し：なし 戻り値の型：Object 型）

close メソッド

説明

このオブジェクトをクローズして，使用中のリソースを解放します。false を返すまで hasMore メソッドや hasMoreElements メソッドを呼び出し続けた場合は，このメソッドを呼び出す必要はありません。

形式

```
public void close()
  throws NamingException;
```

パラメタ

なし

例外

javax.naming.NamingException：

クローズ中に NamingException 例外が発生しました。

戻り値

なし

hasMore メソッド

説明

一覧に次のユーザ ID があるかどうかを判定します。

形式

```
public boolean hasMore()
  throws NamingException;
```

パラメタ

なし

例外

javax.naming.NamingException：

次のユーザ ID があるかどうかを判定中に NamingException 例外が発生しました。

戻り値

true：

次のユーザ ID がありました。

false：

次のユーザ ID がありませんでした。

hasMoreElements メソッド

説明

一覧に次のユーザ ID があるかどうかを判定します。例外が発生した場合は false を返却します。

形式

```
public boolean hasMoreElements();
```

パラメタ

なし

例外

なし

戻り値

true：

次のユーザ ID がありました。

false：

次のユーザ ID がありませんでした。

next メソッド

説明

一覧の次のユーザ ID を取得します。

形式

```
public String next()
  throws NamingException;
```

パラメタ

なし

例外

java.util.NoSuchElementException：

次のユーザ ID がないときにこのメソッドを呼び出しました。

javax.naming.NamingException：

次のユーザ ID を取得中に NamingException 例外が発生しました。

戻り値

次のユーザ ID を返却します。

nextElement メソッド

説明

一覧の次のユーザ ID を取得します。next メソッドとの違いは，NamingException 例外が発生しないことと，戻り値の型は Object 型であることです。戻り値の Object オブジェクトを String にキャストして参照してください。このメソッドを実行中に NamingException 例外が発生した場合は null を返却します。

形式

```
public Object nextElement();
```

パラメタ

なし

例外

java.util.NoSuchElementException：

次のユーザ ID がないときにこのメソッドを呼び出しました。

戻り値

次のユーザ ID を返却します。

15.8 LoginUtil クラス

説明

統合ユーザ管理のセッション内でログインしているユーザの有無を調べます。

LoginUtil クラスのパッケージ名は，com.cosminexus.admin.auth.util です。

形式

```
class LoginUtil
{
  public static boolean check(HttpServletRequest request,
                              HttpServletResponse response);
  public static boolean check(HttpServletRequest request,
                               HttpServletResponse response,
                               String realmName);
}
```

メソッド一覧

メソッド名	機能
check メソッド（形式 1）	セッション内でログインしているユーザの有無を調べます。
check メソッド（形式 2）	セッション内でログインしているユーザの有無を調べます。特定のレルム内でログインしているユーザを調べるために使用します。

注意事項

このクラスの check メソッドを使わなくても，HttpSession にログイン時に生成された Subject を関連づけ，Subject の Principal の有無によってログインの有無を判断できます。この方式で確認する場合は，統合ユーザ管理の機能を使ってセッションを停止しないでください。

check メソッド（形式 1）

説明

セッション内でログインしているユーザの有無を調べます。このセッション内でどれかのレルムでログインしているユーザを検索し，一人でもログインしているユーザが見つかれば true を返却します。

形式

```
public static boolean check(HttpServletRequest request,
                            HttpServletResponse response);
```

パラメタ

request：

JSP/Servlet に渡された HttpServletRequest のリファレンスを指定します。null を指定した場合は NullPointerException 例外が発生します。

response：

JSP/Servlet に渡された HttpServletResponse のリファレンスを指定します。null を指定した場合は NullPointerException 例外が発生します。

例外

java.lang.NullPointerException：

このメソッドのパラメタに null を指定しました。

戻り値

true：

ログインしているユーザが見つかりました。

false：

ログインしているユーザが見つかりませんでした。

check メソッド（形式 2）

説明

セッション内でログインしているユーザの有無を調べます。realmName は，特定のレルム内でログインしているユーザを調べるために使用します。

形式

```
public static boolean check(HttpServletRequest request,
                            HttpServletResponse response,
                            String realmName);
```

パラメタ

request：

JSP/Servlet に渡された HttpServletRequest のリファレンスを指定します。null を指定した場合は NullPointerException 例外が発生します。

response：

JSP/Servlet に渡された HttpServletResponse のリファレンスを指定します。null を指定した場合は NullPointerException 例外が発生します。

realmName：

特定のレルム内でログインしているユーザを調べる場合に指定します。null を指定した場合は NullPointerException 例外が発生します。

例外

java.lang.NullPointerException：

このメソッドのパラメタに null を指定しました。

戻り値

true：

ログインしているユーザが見つかりました。

false：

ログインしているユーザが見つかりませんでした。

15.9 ObjectClassEntry クラス

説明

LDAP ディレクトリサーバに作成するユーザエントリやサブコンテキストのオブジェクトクラスを格納するクラスです。

ObjectClassEntry クラスのパッケージ名は，com.cosminexus.admin.auth.api.repository.ldap です。

形式

```
class ObjectClassEntry
{
public ObjectClassEntry();
public ObjectClassEntry(String[] objectClasses);
public ObjectClassEntry(String subcontext,
                        String[] objectClasses);

public void setObjectClasses(String[] objectClasses);
public String[] getObjectClasses();
public void setSubcontext(String subcontext);
public String getSubcontext();
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
ObjectClassEntry コンストラクタ	ObjectClassEntry クラスのインスタンスを生成します。
getObjectClasses メソッド	setObjectClasses メソッドまたはコンストラクタで指定したオブジェクトクラスを取得します。
getSubcontext メソッド	setSubcontext メソッドまたはコンストラクタで指定したサブコンテキストを取得します。
setObjectClasses メソッド	オブジェクトクラスをオブジェクトに格納します。
setSubcontext メソッド	サブコンテキストをオブジェクトに格納します。

ObjectClassEntry コンストラクタ

説明

インスタンスを生成します。パラメタにオブジェクトクラスを指定すると，ユーザエントリのオブジェクトクラスとしてこのオブジェクトに格納されます。

形式

```
public ObjectClassEntry();

public ObjectClassEntry(String[] objectClasses);
```

```
public ObjectClassEntry(String subcontext,
                        String[] objectClasses);
```

パラメタ

subcontext：

サブコンテキストを指定します。このパラメタを省略した場合や，null や空文字("")を指定した場合は，ユーザエントリを表します。AttributeEntry オブジェクトに指定するサブコンテキストと同じ文字列を指定してください。

objectClasses：

ユーザエントリのオブジェクトクラスを String の配列で指定します。このパラメタを省略した場合や null を指定した場合は，何も格納されません。

例外

なし

getObjectClasses メソッド

説明

setObjectClasses メソッドまたはコンストラクタで指定した値を取得します。値が格納されていない場合に getObjectClasses メソッドを呼び出したときは null が返却されます。

形式

```
public String[] getObjectClasses();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトに格納されている値を返却します。

getSubcontext メソッド

説明

setSubcontext メソッドまたはコンストラクタで指定した値を取得します。値が格納されていない場合に getSubcontext メソッドを呼び出したときは null が返却されます。

形式

```
public String getSubcontext();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトに格納されている値を返却します。

setObjectClasses メソッド

説明

オブジェクトクラスをこのオブジェクト内に格納します。すでに値が格納されている場合に setObjectClasses メソッドを呼び出したときは上書きされます。

形式

```
public void setObjectClasses(String[] objectClasses);
```

パラメタ

objectClasses：

　オブジェクトクラスを String の配列で指定します。

例外

なし

戻り値

なし

setSubcontext メソッド

説明

サブコンテキストをこのオブジェクト内に格納します。すでに値が格納されている場合に setSubcontext メソッドを呼び出したときは値が上書きされます。

形式

```
public void setSubcontext(String subcontext);
```

パラメタ

subcontext：

サブコンテキストを指定します。null や空文字("")を指定した場合はユーザエントリを表します。AttributeEntry オブジェクトに設定するサブコンテキストと同じ文字列を指定してください。

例外

なし

戻り値

なし

15.10 PasswordCryptography インタフェース

説明

入力したパスワードを暗号化するためのインタフェースです。

PasswordCryptography インタフェースのパッケージ名は，com.cosminexus.admin.auth.security です。

形式

```
interface PasswordCryptography
{
  public byte[] encrypt(byte[] plain);
}
```

メソッド一覧

メソッド名	機能
encrypt メソッド	リポジトリに登録されている暗号化形式に合わせてパスワードを暗号化します。

encrypt メソッド

説明

リポジトリに登録されている暗号化形式に合わせてパスワードを暗号化します。

形式

```
public byte[] encrypt(byte[] plain);
```

パラメタ

plain：

ログインモジュールがこのメソッドを呼び出すときに，ユーザが指定したパスワード（平文）が格納されます。

例外

なし

戻り値

暗号化した結果を返却します。

15.11　PasswordUtil クラス

説明

ユーザのパスワードを変更するためのクラスです。

PasswordUtil クラスのパッケージ名は，com.cosminexus.admin.auth.util です。

形式

```
class PasswordUtil
{
  public static void changePassword(String name,
                                    String uid,
                                    String oldPassword,
                                    String newPassword)
    throws LoginException,
           SecurityException;
```

}

メソッド一覧

メソッド名	機能
changePassword メソッド	パスワードを変更します。

changePassword メソッド

説明

パラメタで指定された name，uid および oldPassword で本人の確認をして，その結果に問題がなければ新しいパスワードに変更します。シングルサインオン用認証情報が登録されている場合は，シングルサインオン情報リポジトリの内容も変更されます。

このメソッドは static メソッドです。

形式

```
public static void changePassword(String name,
                                  String uid,
                                  String oldPassword,
                                  String newPassword)
    throws LoginException,
      SecurityException;
```

パラメタ：

name：

認証に使用するログインモジュール（LoginContext）のアプリケーション名（name）を指定します。

uid：

変更するユーザのユーザ ID を指定します。

oldPassword：

変更前のパスワードを指定します。

newPassword：

変更後のパスワードを指定します。

例外

javax.security.auth.login.LoginException：

認証に必要な情報がありません。または，ユーザ ID／パスワードが誤っています。

java.lang.SecurityException：

アクセス権がありません。

戻り値

なし

注意事項

- シングルサインオン用認証情報は，レルム名，暗号鍵ファイル，およびシングルサインオン情報リポジトリにアクセスする情報が定義されている場合だけ変更します。それ以外の状態（シングルサインオン用の定義がない）の場合は変更しません。
- シングルサインオン用認証情報が登録されている場合に，リポジトリで例外が発生したり（NamingException 例外が発生），暗号化に失敗したりしたとき，このメソッドは LoginException 例外で失敗します。このとき，パスワードは元の状態に戻します（ロールバックします）。また，パスワードを元に戻すのに失敗した場合も LoginException 例外が発生します。例外クラスの詳細については，「15.32　例外クラス」を参照してください。
- name で指定したアプリケーションで WebPasswordLoginModule または WebPasswordLDAPLoginModule を使用していない場合，LoginException 例外が発生します。
- LDAP ディレクトリサーバが Active Directory の場合は，WebPasswordLDAPLoginModule を使用しているアプリケーションを name に指定してください。

15.12 Principal インタフェース

説明

WebPasswordLoginModule が認証したときのユーザ ID を参照するときに使用します。この製品の実装クラスでは，java.security.Principal インタフェースを継承して作成されたものを認証した Subject に関連づけます。そのため，Principal を参照する場合は，Subject から Principal を求めて getName メソッドで参照してください。

Principal インタフェースのパッケージ名は，java.security です。

15.13　SSOData クラス

説明

シングルサインオン用認証情報を格納するクラスです。

SSOData クラスのパッケージ名は，com.cosminexus.admin.auth.api.repository.ldap です。

形式

```
class SSOData
{
  public SSOData();

  public void setSecretData(String secretData)
    throws CryptoException, UnsatisfiedLinkError, SecurityException;
  public void setPublicData(String publicData);
  public String getPublicData();
  public Enumeration getMappingRealms();
  public String getMapping(String realm);
  public void setMapping(String realm,
                         String uid);
  public void removeMapping(String realm);
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
SSOData コンストラクタ	SSOData クラスのインスタンスを生成します。
getMapping メソッド	レルム名に対応するユーザ ID を取得します。
getMappingRealms メソッド	レルム名一覧を取得します。
getPublicData メソッド	PublicData を取得します。
removeMapping メソッド	指定したレルムを削除します。
setMapping メソッド	接続先レルム名とユーザ ID を格納します。
setPublicData メソッド	PublicData を格納します。
setSecretData メソッド	SecretData を格納します。

SSOData コンストラクタ

説明

SSOData クラスのインスタンスを生成します。

形式

```
public SSOData();
```

パラメタ

なし

例外

なし

getMapping メソッド

説明

このオブジェクトに格納されているマッピング情報から，レルム名に対応するユーザ ID を取得します。指定したレルム名に対応するユーザ ID がない場合は null が返却されます。

形式

```
public String getMapping(String realm);
```

パラメタ

realm：
　接続先のレルム名を指定します。

例外

なし

戻り値

ユーザ ID を返却します。

getMappingRealms メソッド

説明

このオブジェクトに格納されているマッピング情報から，レルム名一覧を取得します。

レルム名を参照する場合は，まず，このメソッドで取得した Enumeration オブジェクトに対して nextElement メソッドを実行して，Object オブジェクトを取得します。取得した Object オブジェクトを String にキャストして，値を参照してください。

形式

```
public Enumeration getMappingRealms();
```

パラメタ

なし

例外

なし

戻り値

レルム名一覧を格納した Enumeration オブジェクトを返却します。

getPublicData メソッド

説明

このオブジェクトに格納されている PublicData を取得します。値が格納されていない場合に getPublicData メソッドを呼び出したときは null が返却されます。

形式

```
public String getPublicData();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトに格納されている値を返却します。

removeMapping メソッド

説明

このオブジェクトに格納されているマッピング情報から，指定したレルムを削除します。指定したレルム名が格納されていない場合は何もしません。

形式

```
public void removeMapping(String realm);
```

パラメタ

realm：

　接続先のレルム名を指定します。

例外

なし

戻り値

なし

setMapping メソッド

説明

パラメタに指定された接続先レルム名とユーザ ID をこのオブジェクト内に格納します。すでに同じレルム名でユーザ ID が格納されている場合は上書きされます。

形式

```
public void setMapping(String realm,
                       String uid);
```

パラメタ

realm：

　接続先のレルム名を指定します。

uid：

　接続先レルムのユーザ ID を指定します。

例外

なし

戻り値

なし

setPublicData メソッド

説明

パラメタに指定された PublicData をこのオブジェクト内に格納します。すでに値が格納されている場合は上書きされます。

形式

```
public void setPublicData(String publicData);
```

パラメタ

publicData：

PublicData を指定します。

例外

なし

戻り値

なし

setSecretData メソッド

説明

このオブジェクトに SecretData を格納します。格納するとき，SecretData は暗号化されます。すでに SecretData が格納されている場合は上書きされます。

形式

```
public void setSecretData(String secretData);
```

パラメタ

secretData：

SecretData を指定します。

例外

com.cosminexus.admin.auth.CryptoException：

暗号鍵ファイルを読み込めないため，SecretData の暗号化に失敗しました。

java.lang.UnsatisfiedLinkError：

シングルサインオンライブラリの読み込みに失敗しました。

java.lang.SecurityException：

SecurityManager が存在し，SecurityManager の checkRead メソッドでファイルへの読み込みアクセスが拒否されました。

戻り値

なし

15.14 SSODataEvent クラス

説明

シングルサインオン用認証情報の更新内容を格納するクラスです。

SSODataEvent クラスのパッケージ名は，com.cosminexus.admin.auth.api.repository.event です。

形式

```
class SSODataEvent
{
  public SSODataEvent(String uid,
                      String secretData,
                      String publicData,
                      String oldSecretData,
                      String oldPublicData);

  public String getUserId();
  public String getSecretData();
  public String getPublicData();
  public String getOldSecretData();
  public String getOldPublicData();
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
SSODataEvent コンストラクタ	SSODataEvent クラスのインスタンスを生成します。
getOldPublicData メソッド	変更前の PublicData を取得します。
getOldSecretData メソッド	変更前の SecretData を取得します。
getPublicData メソッド	PublicData を取得します。
getSecretData メソッド	SecretData を取得します。
getUserId メソッド	ユーザ ID を取得します。

SSODataEvent コンストラクタ

説明

インスタンスを生成し，パラメタに指定されたユーザ ID，SecretData，PublicData，変更前の SecretData，および変更前の PublicData を格納します。

形式

```
public SSODataEvent(String uid,
                    String secretData,
                    String publicData,
```

```
                    String oldSecretData,
                    String oldPublicData);
```

パラメタ

uid：

ユーザ ID を指定します。

secretData：

SecretData を指定します。

publicData：

PublicData を指定します。

oldSecretData：

変更前の SecretData を指定します。

oldPublicData：

変更前の PublicData を指定します。

例外

なし

getOldPublicData メソッド

説明

このオブジェクトに格納されている変更前の PublicData を取得します。

形式

```
public String getOldPublicData();
```

パラメタ

なし

例外

なし

戻り値

変更前の PublicData を返却します。設定されていない場合は null を返却します。

getOldSecretData メソッド

説明

このオブジェクトに格納されている変更前の SecretData を取得します。

形式

```
public String getOldSecretData();
```

パラメタ

なし

例外

なし

戻り値

変更前の SecretData を返却します。設定されていない場合は null を返却します。

getPublicData メソッド

説明

このオブジェクトに格納されている PublicData を取得します。

形式

```
public String getPublicData();
```

パラメタ

なし

例外

なし

戻り値

PublicData を返却します。設定されていない場合は null を返却します。

getSecretData メソッド

説明

このオブジェクトに格納されている SecretData を取得します。

形式

```
public String getSecretData();
```

パラメタ

なし

例外

なし

戻り値

SecretData を返却します。設定されていない場合は null を返却します。

getUserId メソッド

説明

このオブジェクトに格納されているユーザ ID を取得します。

形式

```
public String getUserId();
```

パラメタ

なし

例外

なし

戻り値

ユーザ ID を返却します。

15.15 SSODataListener インタフェース

説明

シングルサインオン用認証情報が更新されたときに，更新が通知されるシングルサインオン用認証情報リスナクラスが実装しなければならないインタフェースです。シングルサインオン用認証情報の更新に同期して他システムの認証情報を更新したい場合，このインタフェースを実装したクラスを作成してください。作成したクラスのインスタンス（オブジェクト）を LdapSSODataManager オブジェクトに addSSODataListener メソッドで登録してください。

SSODataListener インタフェースのパッケージ名は，
com.cosminexus.admin.auth.api.repository.event です。

SSODataListener インタフェースのメソッドは LdapSSODataManager クラスのメソッドから呼び出されます。このとき，パラメタとして SSODataEvent オブジェクトが渡されます。

SSODataListener インタフェースの各メソッドが呼び出されるタイミング（呼び出す LdapSSODataManager クラスのメソッド）と，パラメタで渡される SSODataEvent オブジェクトに格納される値の内容を，次の表に示します。

表 15-2　SSODataEvent オブジェクトに格納される値

呼び出す LdapSSODataManager クラスのメソッド	呼び出される SSODataListener インタフェースのメソッド	SSODataEvent オブジェクトに格納される値				
		ユーザ ID	SecretData	PublicData	変更前の SecretData	変更前の PublicData
addSSOData メソッド	ssoDataAdded メソッド	○	○	○	－	－
modifySSOData メソッド	ssoDataModified メソッド	○	○	○	○	○
removeSSOData メソッド	ssoDataRemoved メソッド	○	○	○	－	－

（凡例）

○：格納されます。

－：格納されません。

ssoDataAdded メソッド，ssoDataModified メソッド，および ssoDataRemoved メソッドの各メソッドで問題が発生した場合，問題の原因がわかるメッセージが格納された ChangeDataFailedException 例外をスローするように，クラスを作成してください。LdapSSODataManager のメソッド呼び出し元には，それらの例外オブジェクトが格納された SSODataListenerException 例外が発生します。

形式

```
interface SSODataListener extends java.util.EventListener
{
  public void ssoDataAdded(SSODataEvent event)
```

```
    throws ChangeDataFailedException;
  public void ssoDataModified(SSODataEvent event)
    throws ChangeDataFailedException;
  public void ssoDataRemoved(SSODataEvent event)
    throws ChangeDataFailedException;
}
```

メソッド一覧

メソッド名	機能
ssoDataAdded メソッド	シングルサインオン用認証情報が追加されたときに呼び出されます。
ssoDataModified メソッド	シングルサインオン用認証情報が変更されたときに呼び出されます。
ssoDataRemoved メソッド	シングルサインオン用認証情報が削除されたときに呼び出されます。

ssoDataAdded メソッド

説明

シングルサインオン用認証情報が追加されたときに呼び出されます。

形式

```
public void ssoDataAdded(SSODataEvent event)
  throws ChangeDataFailedException;
```

パラメタ

event：

シングルサインオン用認証情報が格納されます。

例外

com.cosminexus.admin.auth.api.repository.event.ChangeDataFailedException：

他システムの認証情報の更新に失敗しました。

戻り値

なし

ssoDataModified メソッド

説明

シングルサインオン用認証情報が変更されたときに呼び出されます。

形式

```
public void ssoDataModified(SSODataEvent event)
  throws ChangeDataFailedException;
```

パラメタ

event：

　シングルサインオン用認証情報が格納されます。

例外

com.cosminexus.admin.auth.api.repository.event.ChangeDataFailedException：

　他システムの認証情報の更新に失敗しました。

戻り値

なし

ssoDataRemoved メソッド

説明

シングルサインオン用認証情報が削除されたときに呼び出されます。

形式

```
public void ssoDataRemoved(SSODataEvent event)
  throws ChangeDataFailedException;
```

パラメタ

event：

　シングルサインオン用認証情報が格納されます。

例外

com.cosminexus.admin.auth.api.repository.event.ChangeDataFailedException：
　他システムの認証情報の更新に失敗しました。

戻り値

なし

15.16 SSODataListenerException クラス

説明

シングルサインオン用認証情報リスナクラスで例外が発生した場合に呼び出される例外クラスです。

SSODataListenerException クラスのパッケージ名は,
com.cosminexus.admin.auth.api.repository.event です。

形式

```
class SSODataListenerException extends UAException
{
  public SSODataListenerException();
  public SSODataListenerException(String msg);

  public void setException(SSODataListener listener,
                           ChangeDataFailedException exception);
  public SSODataListener[] getListeners();
  public ChangeDataFailedException getException(SSODataListener listener);
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
SSODataListenerException コンストラクタ	SSODataListenerException クラスのインスタンスを生成します。
getException メソッド	例外オブジェクトを取得します。
getListeners メソッド	例外に格納されているリスナを取得します。
setException メソッド	例外オブジェクトを格納します。

SSODataListenerException コンストラクタ

説明

パラメタに指定されたエラーメッセージを使用して,SSODataListenerException クラスのインスタンスを生成します。

形式

```
public SSODataListenerException();

public SSODataListenerException(String msg);
```

パラメタ

msg：
　エラーメッセージを指定します。

例外

なし

getException メソッド

説明

このオブジェクトに格納されている例外オブジェクトを取得します。指定したリスナの例外オブジェクトが格納されていない場合は null を返却します。

形式

```
public ChangeDataFailedException getException(SSODataListener listener);
```

パラメタ

listener：
　例外が発生したリスナオブジェクトを指定します。

例外

なし

戻り値

ChangeDataFailedException オブジェクトを返却します。

getListeners メソッド

説明

この例外に格納されているリスナをすべて取得します。

形式

```
public SSODataListener[] getListeners();
```

パラメタ

なし

例外

なし

戻り値

リスナオブジェクトの配列を返却します。

setException メソッド

説明

例外の原因になった例外オブジェクトをこのオブジェクトに格納します。すでに同じリスナの例外が格納されていた場合は上書きします。

形式

```
public void setException(SSODataListener listener,
                         ChangeDataFailedException exception);
```

パラメタ

listener：

　例外が発生したリスナオブジェクトを指定します。

exception：

　リスナで発生した ChangeDataFailedException オブジェクトを指定します。

例外

なし

戻り値

なし

15.17　UserAttributes インタフェース

説明

ユーザ認証後，Subject に関連づけられた属性を取得するためのインタフェースです。

UserAttributes インタフェースのパッケージ名は，com.cosminexus.admin.auth です。

形式

```
interface UserAttributes
{
  public Object getAttribute(String alias)
    throws IllegalStateException;
  public Enumeration getAttributes(String alias)
    throws IllegalStateException;
  public void addAttribute(String alias,
                           Object attr)
    throws IllegalStateException;
  public Enumeration getAttributeNames()
    throws IllegalStateException;
  public void removeAttribute(String alias)
    throws IllegalStateException;
  public int size()
    throws IllegalStateException;
  public Enumeration getAliases()
    throws IllegalStateException;
}
```

メソッド一覧

メソッド名	機能
addAttribute メソッド	Subject に属性を追加します。
getAttribute メソッド	Subject に関連づけられた属性を取得します。
getAttributeNames メソッド	Subject に関連づけられた属性名の一覧を取得します。
getAttributes メソッド	Subject に関連づけられた属性をすべて取得します。
removeAttribute メソッド	Subject に関連づけられた属性を削除します。
size メソッド	Subject に関連づけられた属性の総数を取得します。
getAliases メソッド	推奨されていません。getAttributeNames メソッドを使用してください。

注意事項

このオブジェクトが無効な場合に，各メソッドの呼び出しで java.lang.IllegalStateException 例外が発生します。この例外は，java.lang.RuntimeException 例外を継承しているため，catch および throws に記述しなくてもコンパイルできるので注意してください。

addAttribute メソッド

説明

Subject に属性を追加します。一つの属性に対して，複数の属性値を関連づけることができます。このメソッドを使用して関連づけた属性は，ユーザ管理リポジトリには反映されません。

形式

```
public void addAttribute(String alias,
                         Object attr)
  throws IllegalStateException;
```

パラメタ

alias：

Subject に関連づける属性名を指定します。

attr：

Subject に関連づける属性値を指定します。

例外

java.lang.IllegalStateException：

このオブジェクトが無効な場合に発生します。また，この例外は，java.lang.RuntimeException 例外を継承しているため，catch および throws に記述しなくてもコンパイルできるので注意してください。

この例外は，次の条件で発生します。

- このオブジェクトを持つ Subject が read-only です。
- logout メソッドによってログアウト処理されています。

戻り値

なし

getAttribute メソッド

説明

Subject に関連づけられた属性を取得します。要求元では，返された Object をキャストして値を参照します。同じ属性で複数の値を持つ場合は，最初に見つかった Object を返却します。

形式

```
public Object getAttribute(String alias)
  throws IllegalStateException;
```

パラメタ

alias：

Subject に関連づけられた属性名を指定します。AttributeEntry クラスに属性の別名を指定した場合は，その別名を指定します。

例外

java.lang.IllegalStateException：

このオブジェクトが無効な場合に発生します。また，この例外は，java.lang.RuntimeException 例外を継承しているため，catch および throws に記述しなくてもコンパイルできるので注意してください。

この例外は，次の条件で発生します。

- このオブジェクトを持つ Subject が read-only です。
- logout メソッドによってログアウト処理されています。

戻り値

指定した Subject に関連づけられた属性値を返却します。見つからない場合は null を返却します。

getAttributeNames メソッド

説明

Subject に関連づけられた属性名の一覧を取得します。AttributeEntry クラスに属性の別名を指定した場合は，その別名を返却します。

形式

```
public Enumeration getAttributeNames()
  throws IllegalStateException;
```

パラメタ

なし

例外

java.lang.IllegalStateException：

このオブジェクトが無効である場合に発生します。また，この例外は，java.lang.RuntimeException 例外を継承しているため，catch および throws に記述しなくてもコンパイルできるので注意してください。

この例外は，次の条件で発生します。

- このオブジェクトを持つ Subject が read-only です。
- logout メソッドによってログアウト処理されています。

戻り値

Subject に関連づけられた属性の名称の一覧を返却します。AttributeEntry クラスに属性の別名を指定した場合は，その別名を返却します。

getAttributes メソッド

説明

Subject に関連づけられた属性をすべて取得します。要求元では，Enumeration の nextElement メソッドで Object を取得し，キャストして値を参照します。

形式

```
public Enumeration getAttributes(String alias)
  throws IllegalStateException;
```

パラメタ

alias：

Subject に関連づけられた属性名を指定します。AttributeEntry クラスに属性の別名を指定した場合は，その別名を指定します。

例外

java.lang.IllegalStateException：

このオブジェクトが無効である場合に発生します。また，この例外は，java.lang.RuntimeException 例外を継承しているため，catch および throws に記述しなくてもコンパイルできるので注意してください。

この例外は，次の条件で発生します。

- このオブジェクトを持つ Subject が read-only です。

- logout メソッドによってログアウト処理されています。

戻り値

指定した Subject に関連づけられた属性値を返却します。見つからない場合は null を返却します。

removeAttribute メソッド

説明

Subject に関連づけられた属性を削除します。このメソッドを使用して削除した属性は，ユーザ管理リポジトリには反映されません。複数の属性値が関連づけられていてもすべて削除されます。

形式

```
public void removeAttribute(String alias)
  throws IllegalStateException;
```

パラメタ

alias：

Subject に関連づけられた属性の名称を指定します。AttributeEntry クラスに属性の別名を指定した場合は，その別名を指定します。

例外

java.lang.IllegalStateException：

このオブジェクトが無効な場合に発生します。また，この例外は，java.lang.RuntimeException 例外を継承しているため，catch および throws に記述しなくてもコンパイルできるので注意してください。

この例外は，次の条件で発生します。

- このオブジェクトを持つ Subject が read-only です。
- logout メソッドによってログアウト処理されています。

戻り値

なし

size メソッド

説明

Subject に関連づけられた属性の総数を取得します。

形式

```
public int size()
  throws IllegalStateException;
```

パラメタ

なし

例外

java.lang.IllegalStateException：

このオブジェクトが無効な場合に発生します。また，この例外は，java.lang.RuntimeException 例外を継承しているため，catch および throws に記述しなくてもコンパイルできるので注意してください。

この例外は，次の条件で発生します。

- このオブジェクトを持つ Subject が read-only です。
- logout メソッドによってログアウト処理されています。

戻り値

Subject に関連づけられた属性の総数を返却します。

15.18　UserData クラス

説明

ユーザ情報を格納するクラスです。

UserData クラスのパッケージ名は，com.cosminexus.admin.auth.api.repository.ldap です。

形式

```
class UserData
{
  public UserData();

  public void setPassword(String password);
  public Enumeration getAttributeNames();
  public Object getAttribute(String name);
  public Enumeration getAttributes(String name);
  public void addAttribute(String name,
                           Object attr);
  public void removeAttribute(String name);
  public int size();
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
UserData コンストラクタ	UserData クラスのインスタンスを生成します。
addAttribute メソッド	このオブジェクトに属性値を一つ追加します。
getAttribute メソッド	このオブジェクトに格納されている内容から，指定した属性名に対応する属性値の一つを取得します。
getAttributeNames メソッド	このオブジェクトに格納されている属性名の一覧を取得します。
getAttributes メソッド	このオブジェクトに格納されている内容から，指定した属性名に対応する属性値をすべて取得します。
removeAttribute メソッド	このオブジェクトから属性を削除します。
setPassword メソッド	パスワードをこのオブジェクトに格納します。
size メソッド	このオブジェクトに格納されている属性の総数を取得します。

UserData コンストラクタ

説明

インスタンスを生成します。

形式

```
public UserData();
```

パラメタ

なし

例外

なし

addAttribute メソッド

説明

このオブジェクトに属性値を一つ追加します。一つの属性に対して複数の属性値を関連づけることができます。

形式

```
public void addAttribute(String name,
                         Object attr);
```

パラメタ

name：

属性の名称を指定します。属性に別名がある場合は，その別名を指定します。

attr：

属性の値を指定します。

例外

なし

戻り値

なし

getAttribute メソッド

説明

このオブジェクトに格納されている内容から，指定した属性名に対応する属性値の一つを取得します。属性値が複数の場合は，それらの値のどれか一つを取得します。

形式

```
public Object getAttribute(String name);
```

パラメタ

name：

属性の名称を指定します。属性に別名がある場合は，その別名を指定します。

例外

なし

戻り値

属性値を返却します。見つからない場合は null を返却します。

getAttributeNames メソッド

説明

このオブジェクトに格納されている属性名の一覧を取得します。

形式

```
public Enumeration getAttributeNames();
```

パラメタ

なし

例外

なし

戻り値

属性名の一覧を格納した Enumeration オブジェクトを返却します。

getAttributes メソッド

説明

このオブジェクトに格納されている内容から，指定した属性名に対応する属性値をすべて取得します。要求元では，Enumeration の nextElement メソッドで Object を取得し，キャストすることで値を参照します。

形式

```
public Enumeration getAttributes(String name);
```

パラメタ

name：

属性の名称を指定します。属性に別名がある場合は，その別名を指定します。

例外

なし

戻り値

属性値を格納した Enumeration オブジェクトを返却します。見つからない場合は null を返却します。

removeAttribute メソッド

説明

このオブジェクトから属性を削除します。複数の属性値が関連づけられている場合，すべて削除されます。

形式

```
public void removeAttribute(String name);
```

パラメタ

name：

属性の名称を指定します。属性に別名がある場合は，その別名を指定します。

例外

なし

戻り値

なし

setPassword メソッド

説明

パスワードをこのオブジェクトに格納します。すでに値が格納されている場合は上書きされます。

形式

```
public void setPassword(String password);
```

パラメタ

password：
　パスワードを指定します。

例外

なし

戻り値

なし

size メソッド

説明

このオブジェクトに格納されている属性の総数を取得します。

形式

```
public int size();
```

パラメタ

なし

例外

なし

戻り値

属性の総数を返却します。

15.19 WebCertificateCallback クラス

説明

Web サーバで認証した結果を CallbackHandler からログインモジュールに渡すための実装クラスです。

WebCertificateCallback クラスのパッケージ名は，com.cosminexus.admin.auth.callback です。

形式

```
class WebCertificateCallback implements javax.security.auth.callback.Callback
{
  public WebCertificateCallback(String attrName);

  public void setSubjectID(String name);
  public String getSubjectID();
  public void setRequest(HttpServletRequest req);
  public HttpServletRequest getRequest();
  public void setResponse(HttpServletResponse res);
  public HttpServletResponse getResponse();
  public void setAttributeEntries(AttributeEntry[] aliases);
  public AttributeEntry[] getAttributeEntries();
  public void setTagID(String tid);
  public String getTagID();
  public void setTagEntry(String entry);
  public String getTagEntry();
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
WebCertificateCallback コンストラクタ	WebCertificateCallback クラスのインスタンスを生成します。
getAttributeEntries メソッド	setAttributeEntries メソッドで指定した属性の一覧を格納しているオブジェクトのリファレンスを取得します。
getRequest メソッド	setRequest メソッドで指定した HttpServletRequest のリファレンスを取得します。
getResponse メソッド	setResponse メソッドで指定した HttpServletResponse のリファレンスを取得します。
getSubjectID メソッド	setSubjectID メソッドで指定した DN 名を取得します。
getTagEntry メソッド	setTagEntry で指定した entry を取得します。
getTagID メソッド	setTagID で指定した TagID を取得します。
setAttributeEntries メソッド	パラメタに指定された属性の一覧を格納しているオブジェクトのリファレンスをオブジェクトに保管します。
setRequest メソッド	パラメタに指定された HttpServletRequest のリファレンスをオブジェクトに保管します。
setResponse メソッド	パラメタに指定された HttpServletResponse のリファレンスをオブジェクトに保管します。

コンストラクタ・メソッド名	機能
setSubjectID メソッド	パラメタに指定された DN 名をオブジェクトに保管します。
setTagEntry メソッド	パラメタに指定された login タグの entry 要素を保管します。
setTagID メソッド	パラメタに指定された login タグの id 要素を保管します。

WebCertificateCallback コンストラクタ

説明

WebCertificateCallback クラスのインスタンスを生成します。インスタンスは WebCertificateLoginModule の login メソッドの延長で生成されます。

形式

```
public WebCertificateCallback(String attrName);
```

パラメタ

attrName：

DN 名から分解するときの属性名を指定します。

例外

なし

getAttributeEntries メソッド

説明

setAttributeEntries メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public AttributeEntry[] getAttributeEntries();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getRequest メソッド

説明

setRequest メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public HttpServletRequest getRequest();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getResponse メソッド

説明

setResponse メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public HttpServletResponse getResponse();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getSubjectID メソッド

説明

setSubjectID メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public String getSubjectID();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getTagEntry メソッド

説明

setTagEntry メソッドで指定した内容を取得します。API を使用してログインした場合は null が返却されます。

形式

```
public String getTagEntry();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getTagID メソッド

説明

setTagID メソッドで指定した内容を取得します。API を使用してログインした場合は null が返却されます。

形式

```
public String getTagID();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

setAttributeEntries メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setAttributeEntries(AttributeEntry[] aliases);
```

パラメタ

aliases：
　属性の一覧を格納しているオブジェクト（AttributeEntry の配列）のリファレンスを指定します。

例外

なし

戻り値

なし

setRequest メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setRequest(HttpServletRequest req);
```

パラメタ

req：
　HttpServletRequest のリファレンスを指定します。

例外

なし

戻り値

なし

setResponse メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setResponse(HttpServletResponse res);
```

パラメタ

res：

HttpServletResponse のリファレンスを指定します。

例外

なし

戻り値

なし

setSubjectID メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setSubjectID(String uid);
```

パラメタ

uid：

DN 名を指定します。

例外

なし

戻り値

なし

setTagEntry メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。

形式

```
public void setTagEntry(String tid);
```

パラメタ

tid：

login タグの entry 要素を指定します。

例外

なし

戻り値

なし

setTagID メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。

形式

```
public void setTagID(String tid);
```

パラメタ

tid：

login タグの id 要素を指定します。

例外

なし

戻り値

なし

15.20 WebCertificateHandler クラス

説明

Web サーバで SSL 認証した結果の情報を取得するための実装クラスです。ユーザ認証ライブラリの CallbackHandler です。

WebCertificateHandler クラスのパッケージ名は，com.cosminexus.admin.auth.callback です。

形式

```
class WebCertificateHandler
{
  public WebCertificateHandler(HttpServletRequest request,
                               HttpServletResponse response,
                               AttributeEntry[] aliases)
    throws ParameterError;
  public WebCertificateHandler(HttpServletRequest request,
                               HttpServletResponse response,
                               String aliasesFile)
    throws ParameterError, FormatError, FileNotFoundException,
           IOException, SecurityException;

  public void handle(Callback[] callbacks)
    throws IOException, UnsupportedCallbackException;
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
WebCertificateHandler コンストラクタ	WebCertificateHandler クラスのインスタンスを生成します。
handle メソッド	SSL 認証の結果情報を取得します。

WebCertificateHandler コンストラクタ

説明

WebCertificateHandler クラスのインスタンスを生成します。request および response は，必ず指定します。null が指定されていた場合は，ParameterError 例外が呼び出されます。

形式

```
public WebCertificateHandler(HttpServletRequest request,
                             HttpServletResponse response,
                             AttributeEntry[] aliases)
    throws ParameterError;

public WebCertificateHandler(HttpServletRequest request,
                             HttpServletResponse response,
                             String aliasesFile)
```

```
    throws ParameterError, FormatError, FileNotFoundException,
           IOException, SecurityException;
```

パラメタ

request：

JSP/Servlet 起動時のパラメタをそのまま指定します。

response：

JSP/Servlet 起動時のパラメタをそのまま指定します。

aliases：

認証が成功したときに作成する Credential（UserAttributes）に格納する情報を指定します。取得する情報がない場合は null を指定します。この場合，Credential は作成されません（空の UserAttributes オブジェクトが作成されます）。aliases には AttributeEntry オブジェクトの配列を指定します。指定した内容で必要な情報がない場合は，FormatError 例外が発生します（必須の値が格納されていない，または Format にない値が指定されている場合）。

aliasesFile：

認証が成功したときに作成する Credential（UserAttributes）に格納する情報を指定します。取得する情報がない場合は null を指定します。この場合，Credential は作成されません（空の UserAttributes オブジェクトが作成されます）。aliasesFile にはファイル名を指定します。指定した内容で必要な情報がない場合は，FormatError 例外が発生します（必須の値が格納されていない，または Format にない値が指定されている場合）。

例外

java.io.FileNotFoundException：

ファイルがない，ファイルではなくディレクトリである，または何かの理由で開くことができません（FileInputStream クラスのコンストラクタで例外が発生した場合）。

java.lang.SecurityException：

SecurityManager が存在し，SecurityManager の checkRead メソッドでファイルへの読み込みアクセスが拒否されました。

java.io.IOException：

ファイルの読み込みに失敗しました。

com.cosminexus.admin.common.ParameterError：

HttpServletRequest または HttpServletResponse のリファレンスが指定されていません。

com.cosminexus.admin.common.FormatError：

aliases および aliasesFile に指定された内容で必要な情報がありません。または余分に指定されています。

handle メソッド

説明

Web サーバで SSL 認証した結果の情報を取得して，WebCertificateCallback オブジェクト（Callback の実装クラス）のリファレンスに設定します。

形式

```
public void handle(Callback[] callbacks)
   throws IOException, UnsupportedCallbackException;
```

パラメタ

callbacks：

WebSSOCallback オブジェクトのリファレンスを指定されていた場合，セッション情報を設定して返却します。このクラス以外のクラスが指定されていた場合は，コンストラクタに指定された CallbackHandler の handle メソッドを呼び出します。

例外

java.io.IOException：

HttpServletRequest 内に Web サーバで認証した結果がありません。

javax.security.auth.callback.UnsupportedCallbackException：

サポートしていない callbacks リファレンスが指定されています。

戻り値

callbacks に値を設定して返却します。このメソッドでは，戻り値はありません。

15.21 WebCertificateLoginModule クラス

説明

JAAS のログインモジュールの実装クラスです。標準ログインモジュールです。Web サーバで認証された証明書からユーザ属性を求めます。

WebCertificateLoginModule クラスのパッケージ名は，com.cosminexus.admin.auth.login です。

15.22 WebLogoutCallback クラス

説明

Web アプリケーションにわたってきたユーザ情報を CallbackHandler からログインモジュールに渡すための実装クラスです。

WebLogoutCallback クラスのパッケージ名は，com.cosminexus.admin.auth.callback です。

形式

```
class WebLogoutCallback implements javax.security.auth.callback.Callback
{
  private HttpSession session = null;
  private String userID = null;

  public WebLogoutCallback();
  public void setSession(HttpSession session);
  public String getSession();
  public void setUserID(String userID);
  public String getUserID();
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
WebLogoutCallback コンストラクタ	WebLogoutCallback クラスのインスタンスを生成します。
getSession メソッド	setSession メソッドで指定した HttpSession オブジェクトのリファレンスを取得します。
getUserID メソッド	setUserID メソッドで指定したユーザ ID を取得します。
setSession メソッド	パラメタに指定された HttpSession オブジェクトのリファレンスを保管します。
setUserID メソッド	パラメタに指定されたユーザ ID を保管します。

WebLogoutCallback コンストラクタ

説明

WebLogoutCallback クラスのインスタンスを生成します。標準ログインモジュールに WebLogoutHandler が指定され，かつ logout メソッドが呼ばれた場合に生成されます。

形式

```
public WebLogoutCallback();
```

パラメタ

なし

例外

なし

getSession メソッド

説明

setSession メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public HttpSession getSession();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getUserID メソッド

説明

setUserID メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public String getUserID();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

setSession メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setSession(HttpSession session);
```

パラメタ

session：

HttpSession オブジェクトのリファレンスを指定します。

例外

なし

戻り値

なし

setUserID メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setUserID(String userID);
```

パラメタ

userID：

ユーザ ID を指定します。

例外

なし

戻り値

なし

15.23 WebLogoutHandler クラス

説明

ログアウトするユーザからユーザ ID を取得する JAAS Callback Handler クラスの実装です。

WebLogoutHandler クラスのパッケージ名は，com.cosminexus.admin.auth.callback です。

形式

```
class WebLogoutHandler
{
  public WebLogoutHandler(HttpSession session , String userID) throws ParameterError;
  public void handle(Callback[] callbacks)
  throws java.io.IOException, UnsupportedCallbackException;
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
WebLogoutHandler コンストラクタ	WebLogoutHandler クラスのインスタンスを生成します。
handle メソッド	ユーザ ID を取得します。

WebLogoutHandler コンストラクタ

説明

WebLogoutHandler クラスのインスタンスを生成します。

session パラメタおよび userID パラメタは，必ず指定してください。null が指定されていた場合は，ParameterError 例外が発生します。

形式

```
public WebLogoutHandler(HttpSession session, String userID);
```

パラメタ

session：

JSP/Servlet 起動時のパラメタをそのまま指定します。

userID：

ログアウトするユーザ ID を指定します。

例外

com.cosminexus.admin.common.ParameterError：

HttpSession または String のリファレンスが指定されていません。

handle メソッド

説明

ユーザ ID を取得し，WebLogoutCallback オブジェクト（Callback の実装クラス）のリファレンスに設定して統合ユーザ管理フレームワークが提供するログインモジュールに渡します。

形式

```
public void handle(Callback[] callbacks)
  throws UnsupportedCallbackException;
```

パラメタ

callbacks：

WebLogoutCallback オブジェクトのリファレンスを指定されていた場合，ユーザ情報を設定して返却します。上記以外のオブジェクトのリファレンスが指定されていた場合は，UnsupportedCallbackException 例外が発生します。

例外

java.io.IOException：

HttpServletRequest 内に Web サーバで認証した結果がありません。

javax.security.auth.callback.UnsupportedCallbackException：

サポートしていない callbacks リファレンスが指定されました。

戻り値

callbacks に値を設定して返却します。このメソッドでは，戻り値はありません。

15.24 WebPasswordCallback クラス

説明

Web アプリケーションに渡された認証情報を CallbackHandler からログインモジュールに渡すための実装クラスです。

WebPasswordCallback クラスのパッケージ名は，com.cosminexus.admin.auth.callback です。

形式

```
class WebPasswordCallback implements javax.security.auth.callback.Callback
{
  public static final int GETPW;
  public static final int NOPW;

  public WebPasswordCallback();

  public void setName(String name);
  public String getName();
  public void setPassword(String password);
  public String getPassword();
  public void setRequest(HttpServletRequest req);
  public HttpServletRequest getRequest();
  public void setResponse(HttpServletResponse res);
  public HttpServletResponse getResponse();
  public void setAttributeEntries(AttributeEntry[] aliases);
  public AttributeEntry[] getAttributeEntries();

  public void setOption(int option);
  public int getOption();
  public void setTagID(String tid);
  public String getTagID();
  public void setTagEntry(String entry);
  public String getTagEntry();
}
```

メンバ属性

GETPW：

この Callback オブジェクトにすべての情報を設定することを要求します。

NOPW：

この Callback オブジェクトにユーザ ID／パスワードを除いた情報を設定することを要求します（このとき，url に対して forward／include はしません）。

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
WebPasswordCallback コンストラクタ	WebPasswordCallback クラスのインスタンスを生成します。
getAttributeEntries メソッド	setAttributeEntries メソッドで指定した，属性の一覧が格納されたオブジェクトのリファレンスを取得します。
getName メソッド	setName メソッドで指定したユーザ ID を取得します。

コンストラクタ・メソッド名	機能
getOption メソッド	設定されているオプションを取得します。
getPassword メソッド	setPassword メソッドで指定したパスワードを取得します。
getRequest メソッド	setRequest メソッドで指定した HttpServletRequest のリファレンスを取得します。
getResponse メソッド	setResponse メソッドで指定した HttpServletResponse のリファレンスを取得します。
getTagEntry メソッド	setTagEntry メソッドで指定した entry を取得します。
getTagID メソッド	setTagID メソッドで指定した TagID を取得します。
setAttributeEntries メソッド	パラメタに指定された，属性の一覧が格納されたオブジェクトのリファレンスをオブジェクトに保管します。
setName メソッド	パラメタに指定されたユーザ ID をこのオブジェクトに保管します。
setOption メソッド	CallbackHandler で設定する内容を要求します。
setPassword メソッド	パラメタに指定されたパスワードをこのオブジェクトに保管します。
setRequest メソッド	パラメタに指定された HttpServletRequest のリファレンスをこのオブジェクトに保管します。
setResponse メソッド	パラメタに指定された HttpServletResponse のリファレンスをこのオブジェクトに保管します。
setTagEntry メソッド	CallbackHandler メソッドで設定する内容を要求します。パラメタに指定された login タグの entry 要素を保管します。
setTagID メソッド	CallbackHandler メソッドで設定する内容を要求します。パラメタに指定された login タグの id 要素を保管します。

WebPasswordCallback コンストラクタ

説明

WebPasswordCallback クラスのインスタンスを生成します。インスタンスは WebPasswordLoginModule の login メソッドの延長で生成されます。

形式

```
public WebPasswordCallback();
```

パラメタ

なし

例外

なし

getAttributeEntries メソッド

説明

setAttributeEntries メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public AttributeEntry[] getAttributeEntries();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getName メソッド

説明

setName メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public String getName();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getOption メソッド

説明

設定されているオプションを取り出します。値を保管していない場合は GETPW が返却されます（デフォルトは，すべての情報を設定することを要求します）。

形式

```
public int getOption();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getPassword メソッド

説明

setPassword メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public String getPassword();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getRequest メソッド

説明

setRequest メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public HttpServletRequest getRequest();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getResponse メソッド

説明

setResponse メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public HttpServletResponse getResponse();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getTagEntry メソッド

説明

setTagEntry メソッドで指定した内容を取得します。API を使用してログインした場合は null が返却されます。

形式

```
public String getTagEntry();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getTagID メソッド

説明

setTagID メソッドで指定した内容を取得します。API を使用してログインした場合は null が返却されます。

形式

```
public String getTagID();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

setAttributeEntries メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setAttributeEntries(AttributeEntry[] aliases);
```

パラメタ

aliases：

属性の一覧を格納しているオブジェクト（AttributeEntry の配列）のリファレンスを指定します。

例外

なし

戻り値

なし

setName メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setName(String uid);
```

パラメタ

uid：

ユーザ ID を指定します。

例外

なし

戻り値

なし

setOption メソッド

説明

CallbackHandler で設定する内容を要求します。NOPW が指定された場合，ユーザ ID／パスワードを除いた情報を設定することを CallbackHandler に対して要求します（このとき，url に対して forward／include はしません）。GETPW が指定された場合，この Callback オブジェクトに格納できるすべての情報を設定することを CallbackHandler に対して要求します。

すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setOption(int option);
```

パラメタ

option：

GETPW または NOPW を設定します。

- GETPW
  この Callback オブジェクトにすべての情報を設定することを要求します。
- NOPW
  この Callback オブジェクトにユーザ ID／パスワードを除いた情報を設定することを要求します（このとき，url に対して forward／include はしません）。

例外

なし

戻り値

なし

setPassword メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setPassword(String password);
```

パラメタ

password：
　パスワードを指定します。

例外

なし

戻り値

なし

setRequest メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setRequest(HttpServletRequest req);
```

パラメタ

req：
　HttpServletRequest のリファレンスを指定します。

例外

なし

戻り値

なし

setResponse メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setResponse(HttpServletResponse res);
```

パラメタ

res：

HttpServletResponse のリファレンスを指定します。

例外

なし

戻り値

なし

setTagEntry メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。

形式

```
public void setTagEntry(String tid);
```

パラメタ

tid：

login タグの entry 要素を指定します。

例外

なし

戻り値

なし

setTagID メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。

形式

```
public void setTagID(String tid);
```

パラメタ

tid：

login タグの id 要素を指定します。

例外

なし

戻り値

なし

15.25 WebPasswordHandler クラス

説明

Web ブラウザを介して，ユーザからユーザ ID およびパスワードを取得する JAAS Callback Handler クラスの実装です。

ユーザ ID およびパスワードは，それぞれ HTTP リクエストの com.cosminexus.admin.auth.name パラメタおよび com.cosminexus.admin.auth.password パラメタに設定してください。

WebPasswordHandler クラスのパッケージ名は，com.cosminexus.admin.auth.callback です。

形式

```
class WebPasswordHandler
{
  public WebPasswordHandler(HttpServletRequest request,
                            HttpServletResponse response,
                            AttributeEntry [] aliases,
                            String url,
                            boolean urlforward)
    throws FormatError, ParameterError;
  public WebPasswordHandler(HttpServletRequest request,
                            HttpServletResponse response,
                            String aliasesFile,
                            String url,
                            boolean urlforward)
    throws IOException, SecurityException, FormatError,
    ParameterError;

  public void handle(Callback[] callbacks)
    throws IOException, UnsupportedCallbackException;
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
WebPasswordHandler コンストラクタ	WebPasswordHandler クラスのインスタンスを生成します。
handle メソッド	認証情報を取得します。

WebPasswordHandler コンストラクタ

説明

WebPasswordHandler クラスのインスタンスを生成します。

WebPasswordHandler コンストラクタには，Credential（UserAttributes）に格納するユーザ情報（属性）をメモリで指定する場合と，ファイルで指定する場合の二つの形式があります。なお，request パラメタおよび response パラメタは，必ず指定してください。null が指定されていた場合は，ParameterError 例外が発生します。

形式

```
public WebPasswordHandler(HttpServletRequest request,
                          HttpServletResponse response,
                          AttributeEntry[] aliases,
                          String url,
                          boolean urlforward)
  throws FormatError, ParameterError;

public WebPasswordHandler(HttpServletRequest request,
                          HttpServletResponse response,
                          String aliasesFile,
                          String url,
                          boolean urlforward)
  throws IOException, SecurityException, FormatError,
  ParameterError;
```

パラメタ

request：

JSP/Servlet 起動時のパラメタをそのまま指定します。

response：

JSP/Servlet 起動時のパラメタをそのまま指定します。

aliases：

認証が成功したときに作成する Credential（UserAttributes）に格納する情報を指定します。取得する情報がない場合は null を指定します。このとき，Credential は作成されません（空の UserAttributes オブジェクトが作成されます）。aliases には，AttributeEntry オブジェクトの配列を指定します。指定した内容で必要な情報がない場合は，FormatError 例外が発生します（必須の値が格納されていない，または Format にない値が指定されている場合）。

aliasesFile：

認証が成功したときに作成する Credential（UserAttributes）に格納する情報を指定します。取得する情報がない場合は null を指定します。このとき，Credential は作成されません（空の UserAttributes オブジェクトが作成されます）。aliasesFile にはファイル名を指定します。指定した内容で必要な情報がない場合は，FormatError 例外が発生します（必須の値が格納されていない，または Format にない値が指定されている場合）。

url：

ユーザから認証情報（ユーザ ID／パスワード）を取得するための URL を指定します。URL の指定がある場合，urlforward の指定に従って Login Form を RequestDispatcher オブジェクトに渡します（ユーザに対して入力情報を取得する場合）。この指定が不要であれば null を指定します。また，null の場合は，urlforward の値を参照しません。null の場合で，handle メソッドの延長で認証情報の取得ができない（HttpServletRequest 内に格納されていない）ときは，LoginContext クラスの login メソッドの延長で LoginException 例外が発生します。

urlforward：

URL の表示方法を選択します。指定された URL に対して，true の場合，RequestDispatcher オブジェクトの forward メソッドを呼び出します。false の場合，RequestDispatcher オブジェクトに対して include メソッドを呼び出します。

例外

java.io.FileNotFoundException：

ファイルがない，ファイルではなくディレクトリである，または何かの理由で開くことができません（FileInputStream クラスのコンストラクタで例外が発生した場合）。

java.lang.SecurityException：

SecurityManager が存在し，SecurityManager の checkRead メソッドでファイルへの読み込みアクセスが拒否されました。

java.io.IOException：

ファイルの読み込みに失敗しました。

com.cosminexus.admin.common.ParameterError：

HttpServletRequest または HttpServletResponse のリファレンスが指定されていません。

com.cosminexus.admin.common.FormatError：

aliases および aliasesFile に指定された内容で必要な情報がありません。または余分に指定されています。

handle メソッド

説明

認証情報を取得し，WebPasswordCallback オブジェクト（Callback の実装クラス）のリファレンスに設定してユーザ認証ライブラリのログインモジュールに渡します。

形式

```
public void handle(Callback[] callbacks)
  throws IOException, UnsupportedCallbackException;
```

パラメタ

callbacks：

WebPasswordCallback オブジェクトのリファレンスを指定されていた場合，認証情報を設定して返却します。WebSSOCallback オブジェクトのリファレンスが指定されていた場合は，セッション情報を設定して返却します。上記以外のオブジェクトのリファレンスが指定されていた場合は，UnsupportedCallbackException 例外が発生します。

例外

java.io.IOException：

HttpServletRequest 内にユーザ ID／パスワードの情報がありません。取り出すパラメタ名は，「注意事項」を参照してください。

javax.security.auth.callback.UnsupportedCallbackException：

サポートしていない callbacks リファレンスが指定されていました。

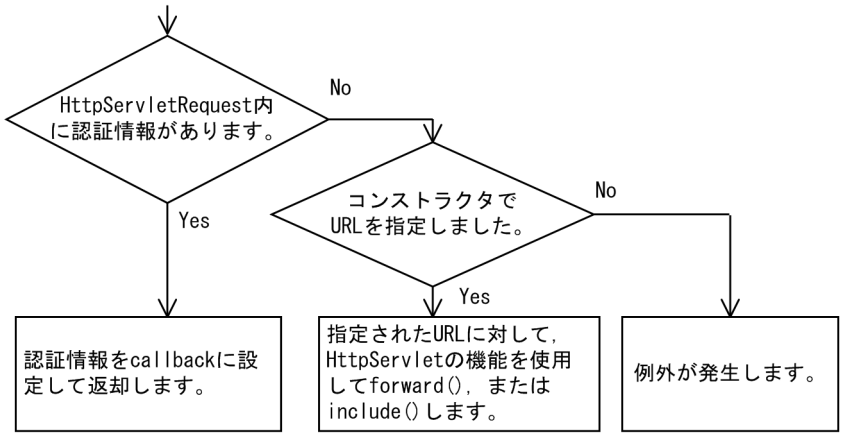
戻り値

callbacks に値を設定して返却します。このメソッドでは，戻り値はありません。

注意事項

認証情報は，次の図に示す順序で読み込まれます。

図 15–1　認証情報の読み込み順序



HttpServletRequest 内の認証情報は，次に示すパラメタ名から取得します。

- com.cosminexus.admin.auth.name
  ユーザが指定したユーザ ID を指定します。
- com.cosminexus.admin.auth.password
  ユーザが指定したパスワードを指定します。

15.26 WebPasswordJDBCLoginModule クラス

説明

JAAS のログインモジュールの実装クラスです。JDBC を使ってデータベースにアクセスし，パスワード認証をします。

WebPasswordJDBCLoginModule クラスのパッケージ名は，com.cosminexus.admin.auth.login です。

15.27 WebPasswordLDAPLoginModule クラス

説明

JAAS のログインモジュールの実装クラスです。LDAP ディレクトリサーバとバインドした結果で認証をします。

WebPasswordLDAPLoginModule クラスのパッケージ名は，com.cosminexus.admin.auth.login です。

15.28 WebPasswordLoginModule クラス

説明

JAAS のログインモジュールの実装クラスです。Web アプリケーションのためのパスワード認証をします。

WebPasswordLoginModule クラスのパッケージ名は，com.cosminexus.admin.auth.login です。

15.29 WebSSOCallback クラス

説明

Web アプリケーションに渡されたセッション情報を CallbackHandler からログインモジュールに渡すための実装クラスです。

WebSSOCallback クラスのパッケージ名は，com.cosminexus.admin.auth.sso.callback です。

形式

```
class WebSSOCallback implements javax.security.auth.callback.Callback
{
  public WebSSOCallback();

  public void setRequest(HttpServletRequest req);
  public HttpServletRequest getRequest();
  public void setResponse(HttpServletResponse res);
  public HttpServletResponse getResponse();
  public void setTagID(String tid);
  public String getTagID();
  public void setTagEntry(String entry);
  public String getTagEntry();

}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
WebSSOCallback コンストラクタ	WebSSOCallback クラスのインスタンスを生成します。
getRequest メソッド	setRequest メソッドで指定した HttpServletRequest のリファレンスを取得します。
getResponse メソッド	setResponse で指定した HttpServletResponse のリファレンスを取得します。
getTagEntry メソッド	setTagEntry メソッドで指定した entry を取得します。
getTagID メソッド	setTagID メソッドで指定した TagID を取得します。
setRequest メソッド	パラメタに指定された HttpServletRequest のリファレンスをオブジェクトに保管します。
setResponse メソッド	パラメタに指定された HttpServletResponse のリファレンスをオブジェクトに保管します。
setTagEntry メソッド	パラメタに指定された login タグの entry 要素を保管します。
setTagID メソッド	パラメタに指定された login タグの id 要素を保管します。

WebSSOCallback コンストラクタ

説明

WebSSOCallback クラスのインスタンスを生成します。インスタンスは WebSSOLoginModule の login メソッドの延長で生成されます。

形式

```
public WebSSOCallback();
```

パラメタ

なし

例外

なし

getRequest メソッド

説明

setRequest メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public HttpServletRequest getRequest();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getResponse メソッド

説明

setResponse メソッドで指定した内容を取得します。値を保管していない場合は null が返却されます。

形式

```
public HttpServletResponse getResponse();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getTagEntry メソッド

説明

setTagEntry メソッドで指定した内容を取得します。API を使用してログインした場合は null が返却されます。

形式

```
public String getTagEntry();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

getTagID メソッド

説明

setTagID メソッドで指定した内容を取得します。API を使用してログインした場合は null が返却されます。

形式

```
public String getTagID();
```

パラメタ

なし

例外

なし

戻り値

このオブジェクトが保管している値を返却します。

setRequest メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setRequest(HttpServletRequest req);
```

パラメタ

req：

HttpServletRequest のリファレンスを指定します。

例外

なし

戻り値

なし

setResponse メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。すでに値を保管している情報がある場合は上書きされます。

形式

```
public void setResponse(HttpServletResponse res);
```

パラメタ

res：

HttpServletResponse のリファレンスを指定します。

例外

なし

戻り値

なし

setTagEntry メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。

形式

```
public void setTagEntry(String tid);
```

パラメタ

tid：

login タグの entry 要素を指定します。

例外

なし

戻り値

なし

setTagID メソッド

説明

パラメタに指定された値をこのオブジェクト内に保管します。

形式

```
public void setTagID(String tid);
```

パラメタ

tid：

login タグの id 要素を指定します。

例外

なし

戻り値

なし

15.30 WebSSOHandler クラス

説明

Web ブラウザを介して，セッションに関する情報を取得する JAAS CallbackHandler クラスの実装です。シングルサインオンライブラリの CallbackHandler です。

このクラスに各システムのログインモジュールに対応した CallbackHandler のリファレンスを指定することで，各システムの CallbackHandler の実装を変更しないでシングルサインオンの機能を実現できます。

WebSSOHandler クラスのパッケージ名は，com.cosminexus.admin.auth.sso.callback です。

形式

```
class WebSSOHandler
{
  public WebSSOHandler(HttpServletRequest request,
                       HttpServletResponse response,
                       CallbackHandler ch)
    throws ParameterError;

  public void handle(Callback[] callbacks)
    throws IOException, UnsupportedCallbackException;
}
```

コンストラクタ・メソッド一覧

コンストラクタ・メソッド名	機能
WebSSOHandler コンストラクタ	WebSSOHandler クラスのインスタンスを生成します。
handle メソッド	セッション情報を取得します。

WebSSOHandler コンストラクタ

説明

WebSSOHandler クラスのインスタンスを生成します。request および response は，必ず指定します。null が指定されていた場合は，ParameterError 例外が発生します。

形式

```
public WebSSOHandler(HttpServletRequest request,
                     HttpServletResponse response,
                     CallbackHandler ch)
  throws ParameterError;
```

パラメタ

request：

JSP/Servlet 起動時のパラメタをそのまま指定します。

response：

JSP/Servlet 起動時のパラメタをそのまま指定します。

ch：

各システムの CallbackHandler のリファレンスを指定します。不要な場合は null を指定します。

例外

com.cosminexus.admin.common.ParameterError：

HttpServletRequest または HttpServletResponse のリファレンスが指定されていません。

handle メソッド

説明

セッション情報を取得して，WebSSOCallback オブジェクト（Callback の実装クラス）のリファレンスに設定してシングルサインオンライブラリのログインモジュールに渡します。

形式

```
public void handle(Callback[] callbacks)
  throws IOException, UnsupportedCallbackException;
```

パラメタ

callbacks：

WebSSOCallback オブジェクトのリファレンスが指定されていた場合，セッション情報を設定して返却します。このクラス以外が指定されていた場合は，コンストラクタに指定された CallbackHandler の handle メソッドを呼び出します。

例外

java.io.IOException：

各システムの CallbackHandler の handle メソッドでこの例外が発生しました。

javax.security.auth.callback.UnsupportedCallbackException：

この例外は，次の条件で発生します。

- コンストラクタに各システムの CallbackHandler のリファレンスを指定していません（null の場合）。

- 各システムの CallbackHandler の handle メソッドでこの例外が発生しました。

戻り値

callbacks に値を設定して返却します。このメソッドでは，戻り値はありません。

15.31 WebSSOLoginModule クラス

説明

JAAS のログインモジュールの実装クラスです。シングルサインオンをするためにほかのログインモジュールを呼び出します。

WebSSOLoginModule クラスのパッケージ名は，com.cosminexus.admin.auth.sso.login です。

15.32　例外クラス

統合ユーザ管理の API で使用する例外クラスについて説明します。使用する例外クラスには，JAAS で規定されているログインモジュールの例外クラスとこの製品で提供する API（JAAS 以外）の例外クラスがあります。

15.32.1　JAAS のログインモジュールの例外クラス

JAAS で規定されているログインモジュールの例外クラスを次の表に示します。

表 15–3　JAAS のログインモジュールの例外クラス一覧

項番	例外名	内容
1	javax.security.auth.login.LoginException	項番 2～4 の親クラスです。コンストラクタのパラメタに msg（java.lang.String）を持ちます。
2	javax.security.auth.login.AccountExpiredException	ユーザアカウントが期限切れであることを通知します。
3	javax.security.auth.login.CredentialExpiredException	Credential が期限切れであることを通知します。
4	javax.security.auth.login.FailedLoginException	認証が失敗したことを通知します。

また，ユーザ認証ライブラリ／シングルサインオンライブラリのログインモジュールでは，これらの例外にエラーメッセージ文字列を設定して送出します。エラーメッセージ文字列の一覧を次の表に示します。

なお，ここで示した例外以外でも，JAAS のコンフィグレーションファイルの記述に誤りがある場合に LoginContext クラスをインスタンス化すると，java.lang.SecurityException 例外が発生します。その場合は，次の表のエラーメッセージ文字列を参照して，JAAS のコンフィグレーションファイルの内容を修正してください。

表 15–4　ユーザ認証ライブラリ／シングルサインオンライブラリのログインモジュールの例外

例外名	エラーメッセージ文字列	発生条件
javax.security.auth.login.FailedLoginException	data not found	認証情報が，渡されたパラメタ内にありません。 WebPasswordHandler クラスに渡した HttpServletRequest 内にユーザ ID／パスワードが格納されていません。
	invalid data	• ユーザ ID／パスワードに誤りがあり，認証できません。 • 証明書から取り出したユーザ ID に対応したエントリが，リポジトリ内にありません。
	no data	該当セッション内で認証済みである場合に，呼び出そうとしたレルムに対応したシングルサインオン用認証情報が定義されていません。

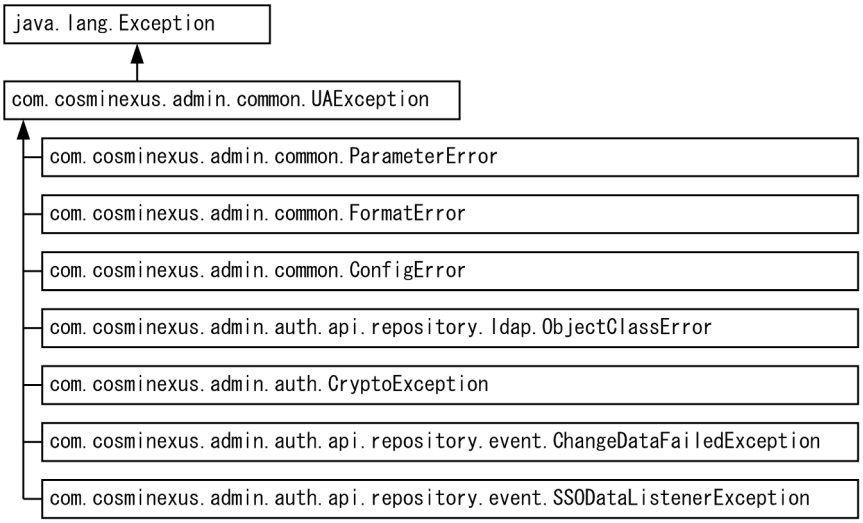
例外名	エラーメッセージ文字列	発生条件
javax.security.auth.login.LoginException	invalid parameter	Credential を作成しようとしたときの属性名と属性の一覧の指定内容に次のような誤りがあります。 • 属性名が指定されていません。 • 同じ Alias が重複して指定されました。
	SQL の例外名称	JDBC を使用したアクセスに失敗しました。この例外が発生した場合は，エラーメッセージ文字列を参照して対処してください。
	JNDI の例外名称	LDAP のアクセスに失敗しました。 • LDAP サーバが見つかりません（CommunicationException）。 • バインド DN の指定ミスです（AuthenticationException）。
	not supported	未サポートの CallbackHandler を使用しました。 • WebSSOLoginModule または WebPasswordLoginModule で必要とする情報が，CallbackHandler で求められません。 • handle メソッドで例外が発生しました。この例外は，上記の条件でユーザ管理が提供する CallbackHandler だけで発生します。
	no class for xxx	WebSSOLoginModule から呼び出すクラスが使用できません（xxx は com.cosminexus.admin.auth.sso.loginmodule で指定された値）。 • インスタンス化ができません。JAAS のログインモジュールを継承していません。また，アクセス権限がない，クラスパスが設定されていない場合があります。
	config error	• JAAS のコンフィグレーションファイルに必要な情報が記述されていないため，処理を続行できませんでした。 • 標準ログインモジュールによるユーザ管理のコンフィグレーションファイルに必要な情報が記述されていないため，処理を続行できませんでした。
	invalid session	HttpSession オブジェクトに関連づけようとしましたが，HttpSession オブジェクトが無効になりました。
	crypto error	暗号化／復号化で失敗しました。 • JNI 機能で呼び出すシングルサインオンライブラリの共用ライブラリが見つかりません（java.library.path の設定に問題があります）。 • 復号化に失敗しました（暗号化と復号化で異なる鍵を使用しています）。
	no sso data	シングルサインオン用の情報がありません。 • シングルサインオンをするために必要な情報が定義されていませんでした。
	no principal	Principal がないため，最初に認証したユーザを特定できませんでした。

例外名	エラーメッセージ文字列	発生条件
	class cast error	リポジトリから取り出した型と統合ユーザ管理のコンフィグレーションファイルに指定された型が不一致です。一致するようにしてください。ua.conf（統合ユーザ管理のコンフィグレーションファイル）の com.cosminexus.admin.auth.ldap.password.encrypt を参照してください。ua.conf ファイルについては，「14.2.2　ua.conf（統合ユーザ管理のコンフィグレーションファイル）」を参照してください。
	not found driver	JDBC を使用しました。 • WebPasswordJDBCLoginModule でドライバが見つかりません。ドライバを正しい場所に格納してください。
	その他	各システムのログインモジュールで発生しました。 • WebSSOLoginModule で，ユーザ認証ライブラリ以外のログインモジュールから発生しました。

15.32.2　この製品で提供する API の例外クラス

この製品で提供する API（JAAS 以外）の例外クラスは，次の図に示す階層を持ちます。

図 15–2　例外クラスの階層



例外クラス一覧を次の表に示します。

表 15–5　この製品で提供する API の例外クラス一覧

項番	例外名	内容
1	com.cosminexus.admin.common.UAException	項番 2～8 の親クラスです。

項番	例外名	内容
2	com.cosminexus.admin.common.ParameterError	各 API のパラメタの指定に誤りがありました。
3	com.cosminexus.admin.common.FormatError	Format を持つものに対する指定に誤りがありました。
4	com.cosminexus.admin.common.ConfigError	コンフィグレーションファイルの内容に誤りがありました。
5	com.cosminexus.admin.auth.api.repository.ldap.ObjectClassError	オブジェクトクラスの指定に誤りがありました。
6	com.cosminexus.admin.auth.CryptoException	暗号化／復号化に失敗しました。
7	com.cosminexus.admin.auth.api.repository.event.ChangeDataFailedException	他システムの認証情報の更新に失敗した場合に，リスナクラスが呼び出されます。
8	com.cosminexus.admin.auth.api.repository.event.SSODataListenerException	他システムの認証情報の更新に失敗した場合に，LdapSSODataManager クラスが呼び出されます。

16 統合ユーザ管理フレームワークで使用するタグライブラリ

この章では，統合ユーザ管理フレームワークで使用する JSP タグライブラリについて説明します。

16.1　タグライブラリのタグの一覧

統合ユーザ管理フレームワークでは，JSP で統合ユーザ管理フレームワークの機能を利用してユーザ認証を実装するための JSP タグライブラリを提供します。

統合ユーザ管理フレームワークの JSP タグライブラリを JSP にインポートするためには，次のコードを JSP に記述します。

```
<%@ taglib uri="http://cosminexus.com/admin/auth/uatags" prefix="ua" %>
```

統合ユーザ管理フレームワークが提供する JSP タグライブラリに含まれるタグの一覧を次の表に示します。

表 16-1　JSP タグライブラリのタグの一覧

タグ名	概要
<ua:attributeEntries>Entries</ua:attributeEntries>タグ	<ua:attributeEntry/>タグと協調して，ログイン時に取得するユーザ属性の一覧を指定します。
<ua:attributeEntry/>タグ	ログイン時に取得する個々のユーザ属性を指定します。
<ua:chpw/>タグ	指定したユーザのパスワードを変更します。
<ua:exception>Body</ua:exception>タグ	例外が発生した場合の処理を Body に記述します。
<ua:getPrincipalName/>タグ	ログインしているユーザの Principal 名（ユーザ ID）を取得または表示します。
<ua:getAttribute/>タグ	ログインしているユーザのユーザ属性値を取得または表示します。
<ua:getAttributes/>タグ	ログインしているユーザのユーザ属性値（Multi-Value）を取得または表示します。
<ua:getAttributeNames/>タグ	ログインしているユーザのユーザ属性名の一覧を取得または表示します。
<ua:login/>タグ	ログインします。
<ua:logout/>タグ	ログアウトします。
<ua:notLogin>Body</ua:notLogin>タグ	ログインしていない場合の処理を Body に記述します。各 JSP ページの先頭に記述することで，その JSP ページを処理する前にログインしているかどうか確認できます。

この章では，各タグのタグ属性について表形式で説明しています。表中の「型」はタグ属性の結果として定義されるスクリプト変数の型を，「C/R」はタグ属性の値を JSP コンパイル時に評価するか（C），実行時に評価するか（R）を示します。

16.2　タグライブラリのタグの詳細

16.2.1　<ua:attributeEntries>Entries</ua:attributeEntries>タグ

(1)　説明

<ua:attributeEntry/>タグと協調して，ログイン時に取得するユーザ属性の一覧を指定します。取得するユーザ属性の一覧は，Entries に<ua:attributeEntry/>タグを複数記述して指定します。

```
<ua:attributeEntries id="ae">
  <ua:attributeEntry attrName="cn" />
  <ua:attributeEntry attrName="sn" />
  ...
</ua:attributeEntries>
```

(2)　タグ属性

タグ属性を次の表に示します。

表 16-2　タグ属性（<ua:attributeEntries>Entries</ua:attributeEntries>タグ）

タグ属性	説明	型	要否	C/R
id="id"	id に AttributeEntry クラスの配列のインスタンスを識別する識別子を指定します。さらに，id はこのインスタンスを参照するスクリプト変数の変数名としても利用されます。このインスタンスは scope タグ属性で指定されたスコープを持ちます。ここで使用する識別子は，すべてのスコープで一意の識別子にする必要があります。	AttributeEntry[]	○	C
scope="scope"	scope に id タグ属性で指定したスクリプト変数のスコープを指定します。指定できる値は，"page"，"request"，"session"，"application"のどれかです。それぞれの値の意味は<jsp:useBean/>タグを参照してください。scope タグ属性を省略した場合は，"page"を仮定します。	－	△	C

（凡例）

○：必要です。

△：任意です。

－：該当しません。

C：タグ属性の値を JSP コンパイル時に評価します。

16.2.2　<ua:attributeEntry/>タグ

(1)　説明

ログイン時に取得する個々のユーザ属性を指定します。詳細については，「16.2.1 <ua:attributeEntries>Entries</ua:attributeEntries>タグ」を参照してください。

(2)　タグ属性

タグ属性を次の表に示します。

表 16-3　タグ属性（<ua:attributeEntry/>タグ）

タグ属性	説明	型	要否	C/R
attrName="attrName"	attrName にログイン時に取得するユーザ属性の名称を指定します。	－	○	R
alias="alias"	alias に取得するユーザ属性の別名を指定します。	－	△	R
subCxt="subCxt"	subCxt にユーザ管理コンテキストからのサブコンテキストを指定します。	－	△	R

（凡例）

○：必要です。

△：任意です。

－：該当しません。

R：タグ属性の値を実行時に評価します。

16.2.3　<ua:chpw/>タグ

(1)　説明

指定したユーザのパスワードを変更します。

(2)　タグ属性

タグ属性を次の表に示します。

表 16-4　タグ属性（<ua:chpw/>タグ）

タグ属性	説明	型	要否	C/R
entry="JAAS entry"	JAAS entry に JAAS コンフィグレーションファイルのエントリ名を指定します。	－	○	R

タグ属性	説明	型	要否	C/R
userid="userid"	userid にパスワードを変更するユーザのユーザ ID を指定します。	－	○※	R
useridParam="useridParam"	useridParam にパスワードを変更するユーザのユーザ ID が格納された HTTP リクエストパラメタの名前を指定します。	－		R
oldpw="oldpw"	oldpw に現在のパスワードを平文で指定します。	－	○※	R
oldpwParam="oldpwParam"	oldpwParam に現在のパスワードが格納された HTTP リクエストパラメタの名前を指定します。	－		R
newpw="newpw"	newpw に新しいパスワードを平文で指定します。	－	○※	R
newpwParam="newpwParam"	newpwParam に新しいパスワードが格納された HTTP リクエストパラメタの名前を指定します。	－		R
excepId="excepId"	excepId にパスワード変更処理中に発生した例外のインスタンスを識別する識別子を指定します。このインスタンスは excepScope タグ属性で指定されたスコープを持ちます。ここで使用する識別子は，すべてのスコープで一意の識別子にする必要があります。excepId タグ属性を省略した場合は，パスワード変更処理中に発生した例外は，JspException 例外として<ua:chpw/>タグの外側に伝わります。	－	△	C
excepScope="scope"	scope に excepId タグ属性で指定したスクリプト変数のスコープを指定します。指定できる値 は，"page"，"request"，"session"，"application"のどれかです。それぞれの値の意味は，<jsp:useBean/>タグを参照してください。 excepScope タグ属性を省略した場合 は，"page"を仮定します。	－	△	C

（凡例）

○：必要です。

△：任意です。

－：該当しません。

C：タグ属性の値を JSP コンパイル時に評価します。

R：タグ属性の値を実行時に評価します。

注※

どちらかを指定します。

16.2.4　<ua:exception>Body</ua:exception>タグ

(1)　説明

指定した例外が発生した場合の処理を Body に記述します。

(2)　タグ属性

タグ属性を次の表に示します。

表 16–5　タグ属性（<ua:exception>Body</ua:exception>タグ）

タグ属性	説明	型	要否	C/R
name="name"	name に<ua:login/>タグや<ua:chpw/>タグなどで指定した例外オブジェクトの識別子（excepId タグ属性で指定）を指定します。誤った識別子を指定した場合は何もしません。	－	○	C
type="type"	type に catch する例外オブジェクトのクラス名を，パッケージ名を含めた完全名で指定します。catch しようとする例外オブジェクトが指定したクラス名のクラスにキャストできる場合，Body が実行されます。type タグ属性を省略した場合は，"java.lang.Throwable"を仮定します。	－	△	C
proceed="proceed"	proceed に Body 処理後，残りの JSP ページを処理するかどうかを指定します。"true"を指定した場合（Ignore Case）は残りの JSP ページを処理し，そうでない場合は残りの JSP ページを処理しません。proceed タグ属性を省略した場合は，"false"を仮定します。	－	△	R

（凡例）

○：必要です。

△：任意です。

－：該当しません。

C：タグ属性の値を JSP コンパイル時に評価します。

R：タグ属性の値を実行時に評価します。

16.2.5　<ua:getPrincipalName/>タグ

(1)　説明

ログインしているユーザの Principal 名（ユーザ ID）を取得または表示します。

(2) タグ属性

タグ属性を次の表に示します。

表 16-6 タグ属性（<ua:getPrincipalName/>タグ）

タグ属性	説明	型	要否	C/R
id="id"	id にログインしているユーザの Principal 名（ユーザ ID）を参照するインスタンスを識別する識別子を指定します。さらに，id はこのインスタンスを参照するスクリプト変数の変数名としても利用されます。このインスタンスはページスコープを持ちます。そのため，同一 JSP ページ内で参照できます。したがって，id には，ページスコープで一意の識別子を指定する必要があります。id タグ属性を省略した場合は，取得した Principal 名を JSP に埋め込みます。	String	△	C
name="name"	name に<ua:login/>タグで指定した JAAS LoginContext オブジェクトの識別子（id タグ属性で指定）を指定します。誤った識別子を指定した場合は，id スクリプト変数に null を設定します。	－	○	R

（凡例）

○：必要です。

△：任意です。

－：該当しません。

C：タグ属性の値を JSP コンパイル時に評価します。

R：タグ属性の値を実行時に評価します。

16.2.6 <ua:getAttribute/>タグ

(1) 説明

ログインしているユーザのユーザ属性値を取得または表示します。

(2) タグ属性

タグ属性を次の表に示します。

表 16-7 タグ属性（<ua:getAttribute/>タグ）

タグ属性	説明	型	要否	C/R
id="id"	id にログインしているユーザのユーザ属性値を参照するインスタンスを識別する識別子を指定	type タグ属性の値	△	C

タグ属性	説明	型	要否	C/R
	します。さらに，id はこのインスタンスを参照するスクリプト変数の変数名としても利用されます。このインスタンスはページスコープを持ちます。そのため，同一 JSP ページ内で参照できます。したがって，id には，ページスコープで一意の識別子を指定する必要があります。id タグ属性を省略した場合は，取得したユーザ属性値を toString メソッドを使用して JSP に埋め込みます。			
name="name"	name に<ua:login/>タグで指定した JAAS LoginContext オブジェクトの識別子（id タグ属性で指定）を指定します。誤った識別子を指定した場合は，id スクリプト変数に null を設定します。	－	○	R
attrName="attrName"	attrName に取得するユーザ属性の属性名を指定します。指定したユーザ属性がない場合，id スクリプト変数に null を設定します。	－	○	R
type="type"	type に取得する属性オブジェクトのクラス名を，パッケージ名を含めた完全名で指定します。type タグ属性を省略した場合は，"java.lang.String"を仮定します。	－	△	C

（凡例）

○：必要です。

△：任意です。

－：該当しません。

C：タグ属性の値を JSP コンパイル時に評価します。

R：タグ属性の値を実行時に評価します。

16.2.7　<ua:getAttributes/>タグ

(1)　説明

ログインしているユーザのユーザ属性値（Multi-Value）を取得または表示します。

(2)　タグ属性

タグ属性を次の表に示します。

表 16–8　タグ属性（<ua:getAttributes/>タグ）

タグ属性	説明	型	要否	C/R
id="id"	id にログインしているユーザのユーザ属性値（Multi-Value）を参照するインスタンスを識別する識別子を指定します。さらに，id はこのインスタンスを参照するスクリプト変数の変数名としても利用されます。このスクリプト変数の型は，java.util. Enumeration です。このインスタンスはページスコープを持ちます。そのため，同一 JSP ページ内で参照できます。したがって，id には，ページスコープで一意の識別子を指定する必要があります。id タグ属性を省略した場合は，取得したユーザ属性値を toString メソッドを使用して JSP に 1 行ずつ埋め込みます。	java.util.Enumeration	△	C
name="name"	name に<ua:login/>タグで指定した JAAS LoginContext オブジェクトの識別子（id タグ属性で指定）を指定します。誤った識別子を指定した場合は，id スクリプト変数に null を設定します。	－	○	R
attrName="attrName"	attrName に取得するユーザ属性の属性名を指定します。指定したユーザ属性がない場合，id スクリプト変数に null を設定します。	－	○	R

（凡例）

○：必要です。

△：任意です。

－：該当しません。

C：タグ属性の値を JSP コンパイル時に評価します。

R：タグ属性の値を実行時に評価します。

16.2.8　<ua:getAttributeNames/>タグ

(1)　説明

ログインしているユーザのユーザ属性名の一覧を取得または表示します。

(2)　タグ属性

タグ属性を次の表に示します。

表 16-9　タグ属性（<ua:getAttributeNames/>タグ）

タグ属性	説明	型	要否	C/R
id="id"	id にログインしているユーザのユーザ属性名の一覧を参照するインスタンスを識別する識別子を指定します。さらに，id はこのインスタンスを参照するスクリプト変数の変数名としても利用されます。このスクリプト変数の型は，java.util.Enumeration です。このインスタンスはページスコープを持ちます。そのため，同一 JSP ページ内で参照できます。したがって，id には，ページスコープで一意の識別子を指定する必要があります。id タグ属性を省略した場合は，取得したユーザ属性名を toString メソッドを使用して JSP に 1 行ずつ埋め込みます。	java.util.Enumeration	△	C
name="name"	name に<ua:login/>タグで指定した JAAS LoginContext オブジェクトの識別子（id タグ属性で指定）を指定します。誤った識別子を指定した場合は，id スクリプト変数に null を設定します。	－	○	R

（凡例）

○：必要です。

△：任意です。

－：該当しません。

C：タグ属性の値を JSP コンパイル時に評価します。

R：タグ属性の値を実行時に評価します。

16.2.9　<ua:login/>タグ

(1)　説明

指定した JAAS コンフィグレーション・エントリを使用してログインします。このとき，HTTP リクエストオブジェクトに，com.cosminexus.admin.auth.name パラメタおよび com.cosminexus.admin.auth.password パラメタが設定されている必要があります。また，<ua:login/>タグに対応した<ua:logout/>タグが記述されていない場合は，セッション切断時に暗黙的にログアウトします。

(2)　タグ属性

タグ属性を次の表に示します。

表 16-10　タグ属性（<ua:login/>タグ）

タグ属性	説明	型	要否	C/R
id="id"	id に JAAS LoginContext クラスのインスタンスを識別する識別子を指定します。さらに，id はこのインスタンスを参照するスクリプト変数の変数名としても利用されます。このインスタンスはセッションスコープを持ちます。つまり，同一セッションに参加する異なる JSP ページで参照できます。ここで使用する識別子は，すべてのスコープで一意の識別子にする必要があります。	javax.security.auth.login.LoginContext	○	C
entry="JAAS entry"	JAAS entry に JAAS コンフィグレーションファイルのエントリ名を指定します。	－	○	R
attrFile="attrFile"	attrFile にユーザ管理リポジトリから取得する属性を記述したファイルのファイル名を指定します。ファイルの形式については，シングルサインオン用認証情報の CSV 形式ファイルを参照してください。シングルサインオン用認証情報の CSV 形式ファイルについては，「14.3　シングルサインオン用認証情報の CSV 形式ファイル」を参照してください。	－	△※	R
attrEntName="attrEntName"	attrEntName に<ua:attributeEntries>タグの id タグ属性で指定した識別子を指定します。	－		R
excepId="excepId"	excepId にログイン処理中に発生した例外のインスタンスを識別する識別子を指定します。このインスタンスは excepScope タグ属性で指定されたスコープを持ちます。ここで使用する識別子は，すべてのスコープで一意の識別子にする必要があります。excepId タグ属性を省略した場合は，ログイン処理中に発生した例外は JspException 例外として<ua:login/>タグの外側に伝わります。	－	△	C
excepScope="scope"	scope に excepId タグ属性で指定したスクリプト変数のスコープを指定します。指定できる値は，"page"，"request"，"session"，"application"のどれかです。それぞれの値の意味は，<jsp:useBean/>タグを参照してください。excepScope タグ属性を省略した場合は，"page"を仮定します。	－	△	C

（凡例）

○：必要です。

△：任意です。

－：該当しません。

C：タグ属性の値を JSP コンパイル時に評価します。

R：タグ属性の値を実行時に評価します。

注※

指定する場合はどちらかを指定します。

16.2.10　<ua:logout/>タグ

(1)　説明

ログアウトします。事前にログインしていない場合は，何もしません。

(2)　タグ属性

タグ属性を次の表に示します。

表 16-11　タグ属性（<ua:logout/>タグ）

タグ属性	説明	型	要否	C/R
name="name"	name に<ua:login/>タグで指定した JAAS LoginContext オブジェクトの識別子（id タグ属性で指定）を指定します。誤った識別子を指定した場合は，何もしません。	－	○	R

（凡例）

○：必要です。

－：該当しません。

R：タグ属性の値を実行時に評価します。

16.2.11　<ua:notLogin>Body</ua:notLogin>タグ

(1)　説明

ログインしていない場合の処理を Body に記述します。各 JSP ページの先頭に記述することで，その JSP ページを処理する前にログインしているかどうかを確認できます。

(2)　タグ属性

タグ属性を次の表に示します。

表 16-12　タグ属性（<ua:notLogin>Body</ua:notLogin>タグ）

タグ属性	説明	型	要否	C/R
realm="realm"	realm にレルム名を指定します。指定したレルムにログインしていない場合，Body を実行し	－	△	R

タグ属性	説明	型	要否	C/R
	ます。realm タグ属性を省略した場合は，どのレルムにもログインしていない場合に Body を実行します。			
proceed="proceed"	proceed に Body 処理後，残りの JSP ページを処理するかどうかを指定します。"true"を指定した場合（Ignore Case）は残りの JSP ページを処理し，そうでない場合は残りの JSP ページを処理しません。proceed タグ属性を省略した場合は，"false"を仮定します。	－	△	R

（凡例）

△：任意です。

－：該当しません。

R：タグ属性の値を実行時に評価します。

17 EJB クライアントアプリケーションの実装で使用する API

この章では，EJB クライアントアプリケーションで使用する API のうち，セキュリティに関するクラスについて説明します。

17.1 LoginInfoManager クラス

説明

J2EE サーバに設定したユーザとパスワードを使って，セキュリティ認証を実行します。

セキュリティ認証を実行する J2EE サーバについて説明します。

- ejbserver.security.service.url プロパティを指定している場合

  セキュリティ認証は，ejbserver.security.service.url プロパティに指定した CORBA ネーミングサービスに接続している J2EE サーバのうち，ejbserver.serverName プロパティで指定したサーバ名称と一致する J2EE サーバで実行されます。

  ejbserver.security.service.url プロパティは，java.naming.provider.url プロパティに指定した CORBA ネーミングサービスに接続していない J2EE サーバでセキュリティ認証を実行する場合に，指定してください。

- ejbserver.security.service.url プロパティを指定していない場合

  セキュリティ認証は，java.naming.provider.url プロパティに指定した CORBA ネーミングサービスに接続している J2EE サーバのうち，ejbserver.serverName プロパティで指定したサーバ名称と一致する J2EE サーバで実行されます。

JNDI ラウンドロビン検索機能や CTM 連携機能などを使った負荷分散構成の場合は，セキュリティ認証を実行できる J2EE サーバが複数存在することになります。この場合，構成する J2EE サーバすべてに同じユーザ，および同じロールの情報を設定した上で，セキュリティ認証用の J2EE サーバを一つ決定してください。

各プロパティの詳細については，マニュアル「アプリケーションサーバ リファレンス 定義編(サーバ定義)」の「12. Java アプリケーションで使用するファイル」を参照してください。EJB クライアントアプリケーションでのセキュリティの実装方法については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(EJB コンテナ)」の「3.6 EJB クライアントアプリケーションでのセキュリティの実装」を参照してください。

LoginInfoManager クラスのパッケージ名は，
com.hitachi.software.ejb.security.base.authentication です。

メソッド一覧

メソッド名	機能
getLoginInfoManager メソッド	LoginInfoManager オブジェクトを取得します。
login メソッド	J2EE サーバにログインします。
logout メソッド	J2EE サーバからログアウトします。

注意事項

LoginInfoManager クラスのメソッドを使用する場合は次の点に注意してください。

- LoginInfoManager クラスのメソッドは，EJB クライアントアプリケーションから発行することを推奨しています。JSP，サーブレット，または EJB 内から発行した場合，RunAs 機能によって設定した情報が，リクエスト単位で削除されます。

- login メソッドを発行して J2EE サーバを呼び出したあとは，必ず logout メソッドを発行してください。
- login メソッドおよび logout メソッドを入れ子で発行しないでください。logout メソッドを発行しないで login メソッドを連続で発行すると，1 回目の login メソッドで指定した情報が 2 回目の login メソッドによって上書きされます。

getLoginInfoManager メソッド

説明

LoginInfoManager オブジェクトを取得します。

形式

```
public static LoginInfoManager getLoginInfoManager();
```

パラメタ

なし

例外

なし

戻り値

LoginInfoManager オブジェクトを返却します。

login メソッド

説明

J2EE サーバにログインします。

形式

```
public final boolean login(String username,
                           String password)
  throws NotFoundServerException,InvalidUserNameException,
         InvalidPasswordException;
```

パラメタ

username：
　ユーザ名（plain text）を指定します。

password：
　パスワード（plain text）を指定します。

例外

com.hitachi.software.ejb.security.base.authentication.NotFoundServerException：
　J2EE サーバが見つかりません。

com.hitachi.software.ejb.security.base.authentication.InvalidUserNameException：
　指定されたユーザ名が見つかりません。

com.hitachi.software.ejb.security.base.authentication.InvalidPasswordException：
　指定したパスワードが不正です。

戻り値

true：
　ログインに成功しました。

false：
　ログインに失敗しました。

logout メソッド

説明

J2EE サーバからログアウトします。

形式

```
public final void logout();
```

パラメタ

なし

例外

なし

戻り値

なし

18 API による直接接続を使用する負荷分散機の制御で使用するファイル

この章では，API による直接接続を使用する負荷分散機を，運用管理機能から制御する場合に使用するファイルの形式，格納先，機能，指定できるキーなどについて説明します。

18.1　API による直接接続を使用する負荷分散機の制御で使用するファイルの一覧

API による直接接続を使用する負荷分散機の制御で使用するファイルの一覧を次の表に示します。

表 18–1　API による直接接続を使用する負荷分散機の制御で使用するファイルの一覧

ファイル名	分類	概要	参照先
lb.properties	負荷分散機定義プロパティファイル	負荷分散機へのアクセスに必要な接続情報を設定します。	18.2.1
<LB 接続情報の識別名>.properties	仮想サーバマネージャ側の負荷分散機接続設定プロパティファイル	仮想サーバマネージャに，負荷分散機へのアクセスに必要な接続情報を設定します。	18.2.2
tierlb.properties	ティア側の負荷分散機接続設定プロパティファイル	負荷分散機へのアクセスに必要な接続情報をティアに設定します。	18.2.3

18.2 APIによる直接接続を使用する負荷分散機の制御で使用するファイルの詳細

18.2.1 lb.properties（負荷分散機定義プロパティファイル）

(1) 形式

Javaプロパティ形式です。

(2) ファイルの格納先

- Windowsの場合

  <Application Serverのインストールディレクトリ>¥manager¥config

- UNIXの場合

  /opt/Cosminexus/manager/config

(3) 機能

負荷分散機へのアクセスに必要な接続情報を設定します。Application Serverから負荷分散機を制御する場合に使用します。

(4) 設定できるキー

APIを使用した直接接続を使用する負荷分散機で設定できるキーとデフォルトを次に示します。

キー名称	内容	デフォルト
lb.list	負荷分散機の管理IPアドレスをドット記法（xxx.xxx.xxx.xxx）で設定します。xxxには0～255の整数を指定します。なお，複数の負荷分散機を使用する場合は，コンマ「,」で区切り，複数の管理IPアドレスを設定します。	なし
lb.connect_type.<IPアドレス>※1※2	負荷分散機への接続形態を設定します。 「API」を指定してください。 何も指定しない場合，jp1_nc(旧VR互換)が指定されます。	jp1_nc
lb.enable_passwd.<IPアドレス>	負荷分散機に設定したPrivileged EXECレベルに必要なパスワードを負荷分散機ごとに設定します。 なお，このプロパティはACOSの場合に設定します。	なし
lb.API.user.<IPアドレス>	負荷分散機にAPIでログインするときのユーザ名を設定します。	なし
lb.API.passwd.<IPアドレス>	負荷分散機にAPIでログインするときのユーザのパスワードを設定します。	なし

キー名称	内容	デフォルト
lb.API.port.<IP アドレス>	負荷分散機のポート番号を設定します。指定できる値は 1～65534 の整数です。範囲外の値を指定した場合，デフォルト値が設定されます。	443
lb.API.cookie_persistence_template.<Web システム名>.<IP アドレス>	負荷分散機で作成した cookie パーシステンステンプレート名を設定します。	なし
lb.API.timeout.<IP アドレス>	API メソッド実行時のタイムアウト時間（単位：秒）を設定します。 このキーで設定した時間内に，API メソッドが完了しない場合は，Smart Composer 機能のコマンド（cmx_build_system など）がタイムアウトエラーで異常終了します。 なお，指定できる値は 1～2147483 の整数です。範囲外の値を指定した場合，デフォルト値が設定されます。	10
lb.API.protocol.<IP アドレス>	負荷分散機との通信で使用するプロトコルを指定します。 • http：通信プロトコルに HTTP を使用します。 • https：通信プロトコルに HTTPS を使用します。	https
javax.net.ssl.trustStore	負荷分散機の接続形態に API を指定した場合に，負荷分散機のサーバ証明書を登録したトラストストアを指定します。Java の仕様に従って指定してください。	cacerts（JDK のデフォルトのトラストストア）
javax.net.ssl.trustStorePassword	負荷分散機の接続形態に API を指定した場合に，負荷分散機のサーバ証明書を登録したトラストストアのパスワードを指定します。Java の仕様に従って指定してください。	cacerts（Java の信頼を得たデフォルトのキーストアが使われる）

注※1

lb.list に指定した負荷分散機の管理 IP アドレスを指定します。

注※2

指定できる値以外の値が指定された場合は，cmx_test_lb コマンド，cmx_build_system コマンド，cmx_delete_system コマンド，cmx_start_target コマンド，および cmx_stop_target コマンドを実行したときにエラーになるので注意してください。

(5) 記述例

```
lb.list=192.168.10.100
lb.enable_passwd.192.168.10.100=adminpw

lb.connect_type.192.168.10.100=API
lb.API.user.192.168.10.100=user01
lb.API.passwd.192.168.10.100=user01pw
#lb.API.port.192.168.10.100=443
#lb.API.cookie_persistence_template.MyWebSystem.192.168.10.100=SC_COOKIE_TEMPNAME
#lb.API.timeout.192.168.10.100=10
javax.net.ssl.trustStore=C:¥¥work¥¥ACOS.keystore
javax.net.ssl.trustStorePassword=keystore_pass
```

(6) 注意事項

- このファイルには，パスワードなどの情報が含まれているため，適切にファイルのアクセス権を設定してください。
- Management Server 起動中に設定ファイルを更新した場合，Management Server の再起動後に更新情報が反映されます。
- このファイルを更新した場合，または Management Server マシンと負荷分散機との接続構成を変更した場合は，cmx_test_lb コマンドを使用して，負荷分散機への接続を確認してください。cmx_test_lb コマンドの詳細については，マニュアル「アプリケーションサーバ リファレンス コマンド編」の「8. Smart Composer 機能で使用するコマンド」を参照してください。負荷分散機へ接続できない場合は，cmx_test_lb コマンドが出力したメッセージを基に，負荷分散機の設定および簡易構築定義ファイルの負荷分散機の定義（<load-balancer>タグの定義）の設定内容を確認してください。負荷分散機の設定については，「8.5 Management Server（Smart Composer 機能）での負荷分散機の接続情報の設定」を，簡易構築定義ファイルの負荷分散機の定義（<load-balancer>タグの定義）については，マニュアル「アプリケーションサーバ システム構築・運用ガイド」の「4.7.5 負荷分散機へ接続する環境を設定する」を参照してください。

18.2.2 <LB 接続情報の識別名>.properties（仮想サーバマネージャ側の負荷分散機接続設定プロパティファイル）

(1) 形式

J2SE のプロパティファイル形式です。

(2) ファイルの格納先

- Windows の場合

  `<Application Serverのインストールディレクトリ>¥manager¥vmi¥config¥lb¥`
- UNIX の場合

  `/opt/Cosminexus/manager/vmi/config/lb/`

(3) 機能

仮想サーバマネージャに，負荷分散機へのアクセスに必要な接続情報を設定します。

(4) 指定できるキー

指定できるキーを次に示します。なお「省略値」とは，キーの指定がない場合に仮定される値です。「VR」とは，キーが導入・変更されたアプリケーションサーバのバージョンです。

キー名称	内容	指定可能値	省略値	VR
lb.type	負荷分散機の種類を指定します。	指定できる値を次に示します。 • BIG-IPv9 • BIG-IPv10.1 • BIG-IPv10.2 • BIG-IPv11 • ACOS	なし	08-53
lb.host	接続する負荷分散機の管理用 IP アドレスを指定します。	IPv4 ドット記法	なし	08-53
lb.protocol	負荷分散機への接続方式を指定します。 API： API を使用して接続します。	指定できる値を次に示します。 • API	なし	08-53
lb.port	負荷分散機で使用するポート番号を指定します。	指定できる値を次に示します。 • API 1～65534	• API：443	08-53
lb.user	負荷分散機に接続するときのユーザ名を指定します。	任意の文字列を指定します。	なし	08-53
lb.password	負荷分散機に接続するときのユーザのパスワードを指定します。	任意の文字列を指定します。	空文字	08-53
lb.persistence.cookie-insert.templatename	負荷分散機で作成した cookie パーシステンステンプレート名を指定します。 このキーの指定は，次の条件を満たす場合に有効になります。 • ＜LB 接続情報の識別名＞.properties または tierlb.properties で lb.type=ACOS および lb.protocol=API を指定 • tier.properties で lb.persistence.method=cookie-insert を指定	英数字またはアンダースコア「_」で指定した 1 文字以上 31 文字以内の文字列を指定します。	なし	08-70
lb.timeout	負荷分散機へのログイン処理時，またはコマンド送信時のタイムアウト時間（単位：秒）を指定します。 このキーで設定した時間内に，負荷分散機へのログイン処理，または負荷分散機に対して発行した CLI コマンドが完了しない場合は，vmiunit コマンドがタイムアウトエラーで異常終了します。	1～2147483	10	08-53

キー名称	内容	指定可能値	省略値	VR
lb.API.protocol	負荷分散機との通信で使用するプロトコルを指定します。 http： 通信プロトコルに HTTP を使用します。 https： 通信プロトコルに HTTPS を使用します。	指定できる値を次に示します。 • http • https	https	09-00
lb.ACOS.privilegedexec.password	ACOS 上に設定した Privileged EXEC レベルに必要なパスワードを指定します。 このキーの指定は，負荷分散機の種類が ACOS の場合だけ有効になります。	任意の文字列を指定します。	なし	08-53
javax.net.ssl.trustStore	負荷分散機のサーバ証明書を登録したトラストストアを指定します。 このキーの指定は，＜LB 接続情報の識別名＞.properties または tierlb.properties で，次に示す値がすべて指定されている場合に，有効になります。 • lb.type=ACOS • lb.protocol=API • lb.API.protocol=https	Java の仕様に従います。	cacerts（JDK のデフォルトのトラストストア）	08-70
javax.net.ssl.trustStorePassword	負荷分散機のサーバ証明書を登録したトラストストアのパスワードを指定します。 このキーの指定は，＜LB 接続情報の識別名＞.properties または tierlb.properties で，次に示す値がすべて指定されている場合に，有効になります。 • lb.type=ACOS • lb.protocol=API • lb.API.protocol=https	Java の仕様に従います。	cacerts（Java の信頼を得たデフォルトのキーストアが使われる）	08-70

(5) 注意事項

- このファイルには，パスワードなどの情報が含まれているため，適切にファイルのアクセス権を設定してください。
- 仮想サーバマネージャ起動中に設定ファイルを更新した場合，仮想サーバマネージャの再起動後に更新情報が反映されます。

18.2.3 tierlb.properties（ティア側の負荷分散機接続設定プロパティファイル）

(1) 形式

J2SE のプロパティファイル形式です。

(2) ファイルの格納先

- Windows の場合
  <定義ディレクトリ>¥<ティア別定義ディレクトリ>¥vmi¥
- UNIX の場合
  <定義ディレクトリ>/<ティア別定義ディレクトリ>/vmi/

次のテンプレートファイルをコピーして利用してください。

- Windows の場合
  <Application Serverのインストールディレクトリ>¥manager¥vmi¥templates¥tierlb.properties
- UNIX の場合
  /opt/Cosminexus/manager/vmi/templates/tierlb.properties

(3) 機能

負荷分散機へのアクセスに必要な接続情報をティアに設定します。

(4) 指定できるキー

指定できるキーを次に示します。なお「省略値」とは，キーの指定がない場合に仮定される値または動作です。「VR」とは，キーが導入・変更されたアプリケーションサーバのバージョンです。

キー名称	内容	指定可能値	省略値	VR
lb.type	負荷分散機の種類を指定します。	指定できる値を次に示します。 • BIG-IPv9 • BIG-IPv10.1 • BIG-IPv10.2 • BIG-IPv11 • ACOS	なし	08-53
lb.host	接続する負荷分散機の管理用 IP アドレスを指定します。	IPv4 ドット記法	なし	08-53

キー名称	内容	指定可能値	省略値	VR
lb.protocol	負荷分散機への接続方式を指定します。 API： API を使用して接続します。	指定できる値を次に示します。 • API	なし	08-53
lb.port	負荷分散機で使用するポート番号を指定します。	指定できる値を次に示します。 • API 1～65534	• API：443	08-53
lb.user	負荷分散機に接続するときのユーザ名を指定します。	任意の文字列を指定します。	なし	08-53
lb.password	負荷分散機に接続するときのユーザのパスワードを指定します。	任意の文字列を指定します。	空文字	08-53
lb.persistence.cookie-insert.templatename	負荷分散機で作成した cookie パーシステンステンプレート名を指定します。 このキーの指定は，次の条件を満たす場合に有効になります。 • ＜LB 接続情報の識別名＞.properties または tierlb.properties で lb.type=ACOS および lb.protocol=API を指定 • tier.properties で lb.persistence.method=cookie-insert を指定	英数字またはアンダースコア「_」で指定した 1 文字以上 31 文字以内の文字列	なし	08-70
lb.timeout	負荷分散機へのログイン処理時，またはコマンド送信時のタイムアウト時間（単位：秒）を指定します。 このキーで設定した時間内に，負荷分散機へのログイン処理，または負荷分散機に対して発行した CLI コマンドが完了しない場合は，vmiunit コマンドがタイムアウトエラーで異常終了します。	1～2147483	10	08-53
lb.API.protocol	負荷分散機との通信で使用するプロトコルを指定します。 http： 通信プロトコルに HTTP を使用します。 https： 通信プロトコルに HTTPS を使用します。	指定できる値を次に示します。 • http • https	https	09-00
lb.ACOS.privilegedexec.password	ACOS 上に設定した Privileged EXEC レベルに必要なパスワードを指定します。 このキーの指定は，負荷分散機の種類が ACOS の場合だけ有効になります。	任意の文字列を指定します。	なし	08-53
javax.net.ssl.trustStore	負荷分散機のサーバ証明書を登録したトラストストアを指定します。	Java の仕様に従う	cacerts（JDK のデフォルト	08-70

キー名称	内容	指定可能値	省略値	VR
	このキーの指定は，＜LB 接続情報の識別名＞.properties または tierlb.properties で，次に示す値がすべて指定されている場合に，有効になります。 • lb.type=ACOS • lb.protocol=API • lb.API.protocol=https		のトラストストア）	
javax.net.ssl.trustStorePassword	負荷分散機のサーバ証明書を登録したトラストストアのパスワードを指定します。 このキーの指定は，＜LB 接続情報の識別名＞.properties または tierlb.properties で，次に示す値がすべて指定されている場合に，有効になります。 • lb.type=ACOS • lb.protocol=API • lb.API.protocol=https	Java の仕様に従う	cacerts（Java の信頼を得たデフォルトのキーストアが使われる）	08-70

19 セキュリティ管理機能で出力されるメッセージ

この章では，セキュリティ管理機能で出力されるメッセージについて説明します。

19.1　メッセージの記述形式

この章でのメッセージの記述形式を次に示します。

XXXXXnnnn-Y

メッセージテキスト

意味

英文のメッセージの意味

要因

メッセージが出力された要因

対処

ユーザが実施する対処

次に，各項目について説明します。

XXXXXnnnn

メッセージ ID を表します。

メッセージ ID を構成する要素について，次に説明します。

XXXXX

メッセージを出力した機能を示す ID（プリフィックス）を表します。Web サービスセキュリティ機能で出力されるメッセージのプリフィックスを次に示します。

- KDCGF
  SOAP メッセージを受信中にエラーが発生した場合に出力されます。
- KDCGK
  共通鍵生成コマンドの処理が正常に終了した場合，または実行中にエラーが発生した場合に出力されます。
- KDCGS
  Web サービスセキュリティ機能定義ファイルの読み込み中にエラーが発生した場合に出力されます。
- KDCGW
  SOAP メッセージを送信中にエラーが発生した場合に出力されます。
- KEOS
  Manager を使用した構築・運用・保守でエラーが発生した場合に出力されます。
- KEXS
  XML Security - Core でエラーが発生した場合に出力されます。

nnnn

メッセージを出力したプログラムで管理するメッセージ番号を表します。それぞれのメッセージには 4 けたの固有の番号が付いています。

Y

メッセージの種別を表します。メッセージの種別は英字 1 文字で示します。

メッセージ種別には次の種類があります。

- E

  エラーが発生した場合に出力されるメッセージであることを示します。このメッセージ種別を持つメッセージが出力された場合の対処方法については，19.2 以降のメッセージの**対処**を参照してください。

- I

  処理が終了したことを通知するメッセージであることを示します。このメッセージ種別を持つメッセージが出力された場合，処理は正常に終了しているので，対処は必要ありません。

- W

  警告を通知するメッセージであることを示します。このメッセージ種別を持つメッセージが出力された場合の対処方法については，19.2 以降のメッセージの**対処**を参照してください。

メッセージテキスト

Web サービスセキュリティ機能で出力されるメッセージのメッセージテキストを示します。なお，Web サービスセキュリティ機能のメッセージは，英文で出力されます。

意味

英文のメッセージの意味を説明します。

要因

メッセージが出力された要因を示します。

対処

ユーザが実施する対処方法を示します。

19.2 KDCGF で始まるメッセージ

Web サービスセキュリティ機能で出力される KDCGF0001 から KDCGF9999 までのメッセージについて説明します。

KDCGF で始まるメッセージは，SOAPFault 形式で出力されます。SOAPFault 形式のメッセージには，次に示す四つの項目があります。

FaultCode

FaultCode が出力されます。FaultCode は，名前空間 URI とローカル部で構成されます。KDCGF で始まるメッセージの FaultCode の名前空間 URI には，「{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}」が出力されます。ローカル部には，エラーの要因を示す文字列が出力されます。

FaultCode の値は，次の方法で取得できます。

- サーバ側の場合

  SOAP 1.1 の場合は，SOAP Fault メッセージの faultcode 要素から FaultCode を取得できます。

  SOAP 1.2 の場合は，SOAP Fault メッセージの soapenv12:Subcode 要素（soapenv12:Code 要素の子要素）に含まれる soapenv12:Value 要素から FaultCode を取得できます。なお，soapenv12:Code 要素の soapenv12:Vaule 要素の値は，soapenv12:Sender です。

- クライアント側の場合

  SOAP 通信基盤が提供する C4Fault クラス，または JAX-WS 機能が提供する javax.xml.ws.soap.SOAPFaultException クラスを使用して FaultCode を取得できます。

SOAP 通信基盤が提供する API の仕様については，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」の「13. SOAP 通信基盤が提供する API」を参照してください。

JAX-WS 機能でフォルトをバインディングする方法については，マニュアル「アプリケーションサーバ Web サービス開発ガイド」の「10.2 JAX-WS エンジンの動作」を参照してください。

FaultString

メッセージ ID およびメッセージの本文が出力されます。メッセージ ID の見方については，「19.1 メッセージの記述形式」を参照してください。

FaultString は次の方法で取得できます。

- サーバ側の場合

  SOAP 1.1 の場合は，SOAP Fault メッセージの faultstring 要素から FaultString を取得できます。

  SOAP 1.2 の場合は，SOAP Fault メッセージの soapenv12:Reason 要素の soapenv12:Text 要素から FaultString を取得できます。

- クライアント側の場合

  SOAP 通信基盤が提供する C4Fault クラス，または JAX-WS 機能が提供する javax.xml.ws.soap.SOAPFaultException クラスを使用して FaultString を取得できます。

FaultActor

Fault の生成者が出力されます。

FaultActor は次の方法で取得できます。

- サーバ側の場合

  SOAP 1.1 の場合は，SOAP Fault メッセージの faultactor 要素から FaultActor を取得できます。

  SOAP 1.2 の場合は，SOAP Fault メッセージの soapenv12:Role 要素から FaultActor を取得できます。

- クライアント側の場合

  SOAP 通信基盤が提供する C4Fault クラス，または JAX-WS 機能が提供する javax.xml.ws.soap.SOAPFaultException クラスを使用して FaultActor を取得できます。

FaultDetails

Fault の詳細が出力されます。

FaultDetails は次の方法で取得できます。

- サーバ側の場合

  SOAP 1.1 の場合は，SOAP Fault メッセージの detail 要素から FaultDetails を取得できます。

  SOAP 1.2 の場合は，SOAP Fault メッセージの soapenv12:Detail 要素から FaultDetails を取得できます。

- クライアント側の場合

  SOAP 通信基盤が提供する C4Fault クラス，または JAX-WS 機能が提供する javax.xml.ws.soap.SOAPFaultException クラスを使用して FaultDetails を取得できます。

KDCGF0001-E

FaultCode：{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}UnsupportedSecurityToken

FaultString：KDCGF0001-E An unsupported security token was specified. (location = <発生場所>)

FaultActor：なし

FaultDetails：なし

意味

<発生場所>でサポートされていないセキュリティトークン要素が使用されています。<発生場所>には次の内容が出力されます。

- Server：サーバ側で受信したメッセージでエラーが発生した場合
- Client：クライアント側で受信したメッセージでエラーが発生した場合

要因

次のうちのどれかがエラーの要因と考えられます。

- BinarySecurityToken 要素の EncodingType 属性が指定されているにもかかわらず，属性値が「http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary」ではない。
- BinarySecurityToken 要素の ValueType 属性が指定されているにもかかわらず，属性値が「http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3」ではない。
- KeyIdentifier 要素に EncodingType 属性が指定されているにもかかわらず，属性値が「http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary」ではない。
- KeyIdentifier 要素に ValueType 属性が指定されているにもかかわらず，属性値が「http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509SubjectKeyIdentifier」ではない。
- WS-Security の Reference 要素に ValueType 属性が指定されているにもかかわらず，属性値が「http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3」ではない。
- WS-Security の SecurityTokenReference 要素の子要素に，Reference 要素または KeyIdentifier 要素以外の要素が指定されている。
- WS-Security の Security 要素の子要素に，XML 暗号の Reference 要素が指定されている。

対処

「要因」に示した内容に該当する SOAP メッセージを送信していないかどうか，メッセージの送信者に確認してください。

KDCGF0002-E

FaultCode：{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}UnsupportedAlgorithm

FaultString：KDCGF0002-E An unsupported signature or encryption algorithm was specified. (location = ＜発生場所＞)

FaultActor：なし

FaultDetails：なし

意味

<発生場所>でサポートされていない署名アルゴリズム，または暗号アルゴリズムが使用されています。<発生場所>には次の内容が出力されます。

- Server：サーバ側で受信したメッセージでエラーが発生した場合
- Client：クライアント側で受信したメッセージでエラーが発生した場合

要因

次のうちのどれかがエラーの要因と考えられます。

- Canonicalization 要素の Algorithm 属性にサポートされていないアルゴリズムが指定されている。
- SignatureMethod 要素の Algorithm 属性にサポートされていないアルゴリズムが指定されている。
- Transform 要素の Algorithm 属性にサポートされていないアルゴリズムが指定されている。
- Canonicalization 要素の Algorithm 属性に Web サービスセキュリティ方針定義ファイルで設定されていないアルゴリズムが指定されている。
- SignatureMethod 要素の Algorithm 属性に Web サービスセキュリティ方針定義ファイルに設定されていないアルゴリズムが指定されている。
- Transform 要素の Algorithm 属性に Web サービスセキュリティ方針定義ファイルで設定されていないアルゴリズムが指定されている。
- XML 暗号の EncryptionMethod 要素の Algorithm 属性にサポートされていないアルゴリズムが指定されている。
- XML 暗号の EncryptionMethod 要素の Algorithm 属性に Web サービスセキュリティ方針定義ファイルで設定されていないアルゴリズムが指定されている。

対処

「要因」に示した内容に該当する SOAP メッセージを送信していないかどうか，メッセージの送信者に確認してください。または，Web サービスセキュリティ方針定義ファイルの設定を見直してください。

Web サービスセキュリティ機能がサポートしているアルゴリズムおよび Web サービスセキュリティ方針定義ファイルの設定については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.2　Web サービスセキュリティ方針定義ファイル」を参照してください。

KDCGF0003-E

FaultCode：{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}InvalidSecurity

FaultString：KDCGF0003-E An error occurred during security header processing. (location = ＜発生場所＞)

FaultActor：なし

FaultDetails：なし

意味

<発生場所>のセキュリティヘッダ内でエラーが発生しました。<発生場所>には次の内容が出力されます。

- Server：サーバ側で受信したメッセージでエラーが発生した場合
- Client：クライアント側で受信したメッセージでエラーが発生した場合

要因

次のうちのどれかがエラーの要因と考えられます。

- Web サービスセキュリティ方針定義ファイルの Timestamp 要素で Created 要素を要求する指定がされているにもかかわらず，受信した SOAP メッセージに Created 要素がない。

- Web サービスセキュリティ方針定義ファイルの Timestamp 要素で Expires 要素を要求する指定がされているにもかかわらず，受信した SOAP メッセージに Expires 要素がない。
- 受信した SOAP メッセージの Created 要素および Expires 要素の ValueType 属性が xsd:dateTime と異なる。
- 値が必要な要素（Created 要素，Expires 要素，BinarySecurityToken 要素，KeyIdentifier 要素）に値がない。
- Web サービスセキュリティ方針定義ファイルで BinarySecurityToken 要素を要求する指定がされているにもかかわらず，受信した SOAP メッセージに BinarySecurityToken 要素がない。
- Reference 要素に URI 属性が付与されていない。
- Reference 要素の URI 属性に値が設定されていない。
- Web サービスセキュリティ方針定義ファイルで SOAP ボディに署名を要求する指定がされているにもかかわらず，受信した SOAP メッセージの SOAP ボディに署名がない。
- 暗号化された SOAP メッセージに KeyInfo 要素がない。
- 暗号化された SOAP メッセージの KeyName 要素で示された鍵が Web サービスセキュリティ機能定義ファイルに記述されていない。
- Web サービスセキュリティ方針定義ファイルで SOAP ボディの要素の暗号化を要求する指定がされているにもかかわらず，受信した SOAP メッセージの SOAP ボディの要素が暗号化されていない。
- 受信した SOAP メッセージに同じ属性値を持つ Id 属性がある。
- Web サービスセキュリティ方針定義ファイルの ReceiverPortConfig 要素の Name 属性と My_role 属性に対応した Web サービスセキュリティ機能定義ファイルの設定がない。

対処

「要因」に示した内容に該当する SOAP メッセージを送信していないかどうか，メッセージの送信者に確認してください。または，Web サービスセキュリティ方針定義ファイルの設定を見直してください。

Web サービスセキュリティ機能がサポートしているアルゴリズムおよび Web サービスセキュリティ方針定義ファイルの設定については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.2　Web サービスセキュリティ方針定義ファイル」を参照してください。

KDCGF0004-E

FaultCode：{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}InvalidSecurityToken

FaultString：KDCGF0004-E An invalid security token was specified (location = <発生場所>)

FaultActor：なし

FaultDetails：なし

意味

<発生場所>で不正なセキュリティトークンが使用されています。<発生場所>には次の内容が出力されます。

- Server：サーバ側で受信したメッセージでエラーが発生した場合
- Client：クライアント側で受信したメッセージでエラーが発生した場合

要因

次のうちのどちらかがエラーの要因と考えられます。

- BinarySecurityToken 要素の ValueType 属性が付与されていない。
- 受信した SOAP メッセージの BinarySecurityToken 要素を，Web サービスセキュリティ方針定義ファイルに記述された証明書ファイルで検証した場合に，常に検証が失敗する。

対処

「要因」に示した内容に該当する SOAP メッセージを送信していないかどうか，メッセージの送信者に確認してください。または，Web サービスセキュリティ方針定義ファイルの設定を見直してください。

Web サービスセキュリティ機能がサポートしているアルゴリズムおよび Web サービスセキュリティ方針定義ファイルの設定については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.2　Web サービスセキュリティ方針定義ファイル」を参照してください。

KDCGF0005-E

FaultCode：{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}FailedAuthentication

FaultString：KDCGF0005-E A security token could not be authenticated or authorized. (location = ＜発生場所＞)

FaultActor：なし

FaultDetails：なし

意味

<発生場所>のセキュリティトークンは，認証または認可できません。<発生場所>には次の内容が出力されます。

- Server：サーバ側で受信したメッセージでエラーが発生した場合
- Client：クライアント側で受信したメッセージでエラーが発生した場合

要因

マニュアル「アプリケーションサーバ メッセージ(構築／運用／開発用)」の「KDCGJ0001-E」の要因を参照してください。

対処

マニュアル「アプリケーションサーバ メッセージ(構築／運用／開発用)」の「KDCGJ0001-E」の対処を参照してください。

KDCGF0006-E

FaultCode：{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}FailedCheck

FaultString：KDCGF0006-E A signature or decryption was invalid. (location = ＜発生場所＞)
FaultActor：なし
FaultDetails：なし

意味

<発生場所>の署名または復号化が不正です。<発生場所>には次の内容が出力されます。

- Server：サーバ側で受信したメッセージでエラーが発生した場合
- Client：クライアント側で受信したメッセージでエラーが発生した場合

要因

次のうちのどちらかがエラーの要因と考えられます。

- 受信した SOAP メッセージに不正な署名が付与されている。
- 受信した SOAP メッセージが不正に暗号化されている。

対処

「要因」に示した内容に該当する SOAP メッセージを送信していないかどうか，メッセージの送信者に確認してください。

KDCGF0007-E

FaultCode：{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}SecurityTokenUnavailable
FaultString：KDCGF0007-E A referenced security token cannot be found. (location = ＜発生場所＞)
FaultActor：なし
FaultDetails：なし

意味

<発生場所>で受信した SOAP メッセージの中で，参照先で示されるセキュリティトークン要素が見つかりませんでした。<発生場所>には次の内容が出力されます。

- Server：サーバ側で受信したメッセージでエラーが発生した場合
- Client：クライアント側で受信したメッセージでエラーが発生した場合

要因

- WS-Security の Reference 要素で指定されている BinarySecurityToken 要素が見つかりません。
- WS-Security の KeyIdentifier 要素で指定されているサブジェクトキー識別子を持つ X.509 証明書が Web サービスセキュリティ機能定義ファイルの VerificationKeyStore 要素で指定しているキーストアファイルの中から見つかりません。

対処

「要因」に示した内容に該当するSOAPメッセージを送信していないかどうか，メッセージの送信者に確認してください。

19.3　KDCGK で始まるメッセージ

Web サービスセキュリティ機能で出力される KDCGK0001 から KDCGK9999 までのメッセージについて説明します。

KDCGK0001-I

Generation of a secret key has finished. (file = <ファイル名>)

意味

共通鍵の生成が完了しました。

<ファイル名>には，生成した共通鍵のファイル名が出力されます。

KDCGK0010-E

An argument is not specified. (argument = <引数>)

意味

引数が指定されていません。

<引数>には，指定する必要がある引数の名称が出力されます。

要因

共通鍵生成コマンドで指定する必要がある引数が指定されていません。

対処

引数を指定してから，再度，共通鍵生成コマンドを実行してください。

共通鍵生成コマンドで指定する引数については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「4.1.1　共通鍵生成コマンド（CWSSCreateSecretKey）」を参照してください。

KDCGK0011-E

An invalid argument is specified. (argument = <引数>)

意味

不正な引数が指定されています。

<引数>には，不正と判定された引数の名称が出力されます。

要因

共通鍵生成コマンドで使用できない，不正な引数が指定されています。

対処

不正と判定された引数を削除してから，再度，共通鍵生成コマンドを実行してください。

共通鍵生成コマンドで指定できる引数については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「4.1.1　共通鍵生成コマンド（CWSSCreateSecretKey）」を参照してください。

KDCGK0012-E

An invalid argument value is specified. (argument = <引数>, value = <引数値>)

意味

引数に不正な値が指定されています。

<引数>と<引数値>には，それぞれ次の内容が出力されます。

<引数>

引数の名称が出力されます。

<引数値>

引数に指定されている値が出力されます。

要因

共通鍵生成コマンドの引数に不正な値が指定されています。

対処

共通鍵生成コマンドの引数，および引数に指定されている値が正しいかどうかを確認してから，再度，共通鍵生成コマンドを実行してください。

共通鍵生成コマンドの引数に指定できる値については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「4.1.1　共通鍵生成コマンド（CWSSCreateSecretKey）」を参照してください。

KDCGK0013-E

A specified file already exists. (file = <ファイル名>)

意味

指定した共通鍵のファイルはすでに存在します。

要因

共通鍵生成コマンドの-o オプションで指定したファイル名のファイルがすでに存在しています。

対処

すでに存在しているファイルとは異なるファイル名を指定して，再度，共通鍵生成コマンドを実行してください。

共通鍵生成コマンドで使用できるオプションについては，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「4.1.1　共通鍵生成コマンド（CWSSCreateSecretKey）」を参照してください。

KDCGK0100-E

An error occurred during output of the key to a file. (details = <詳細>)

意味

ファイルを出力するときにエラーが発生しました。

<詳細>には，エラー内容の詳細が出力されます。

要因

共通鍵生成コマンドで生成した共通鍵をファイルに出力するときにエラーが発生しました。

対処

<詳細>に表示されるエラーの要因を解決してから，再度，共通鍵生成コマンドを実行してください。
<詳細>に表示されたエラーの要因がわからない場合は，システム管理者に連絡してください。

KDCGK0101-E

An error occurred during key creation. (details = <詳細>)

意味

鍵を生成するときにエラーが発生しました。

<詳細>には，エラー内容の詳細が出力されます。

要因

共通鍵生成コマンドで共通鍵を生成するときにエラーが発生しました。

対処

<詳細>に表示されるエラーの要因を解決してから，再度，共通鍵生成コマンドを実行してください。
<詳細>に表示されたエラーの要因がわからない場合は，システム管理者に連絡してください。

KDCGK9000-E

An unexpected error occurred during processing. (details = <詳細>)

意味

処理中に予期しないエラーが発生しました。

<詳細>には，エラー内容の詳細が出力されます。

要因

共通鍵生成コマンドを実行中に原因不明のエラーが発生しました。

対処

システム管理者に連絡してください。

19.4　KDCGS で始まるメッセージ

Web サービスセキュリティ機能で出力される KDCGS0001 から KDCGS9999 までのメッセージについて説明します。

KDCGS0005-E

A URI or TargetId attribute value must start with #. (tag name = <要素名>, attribute name = <属性名>)

意味

URI 属性または TargetId 属性の先頭は「#」でなければなりません。

<要素名>および<属性名>には，それぞれ次の内容が出力されます。

<要素名>

指定されている値の先頭が「#」ではない属性を持つ要素の名称が出力されます。

<属性名>

指定されている値の先頭が「#」ではない属性の名称が出力されます。

要因

Web サービスセキュリティ機能定義ファイルの属性のうち，指定されている値の先頭が「#」ではない属性があります。

対処

Web サービスセキュリティ機能定義ファイルを修正して，<属性名>の属性に指定する値の先頭に「#」を付けてください。

Web サービスセキュリティ機能定義ファイルの属性に指定する値については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.1　Web サービスセキュリティ機能定義ファイル」を参照してください。

KDCGS0007-E

The specified secret key file does not match the KeyType attribute. (secret key file = <共通鍵ファイル名>, KeyType = <アルゴリズム識別子>)

意味

共通鍵のファイルと keytype 属性での指定が一致しません。

<共通鍵ファイル名>および<アルゴリズム識別子>には，それぞれ次の内容が出力されます。

<共通鍵ファイル名>

共通鍵のファイル名が出力されます。

<アルゴリズム識別子>

keytype 属性で指定されたアルゴリズム識別子が出力されます。

要因

Web サービスセキュリティ機能定義ファイルの SecretKeyFile 要素の keytype 属性に指定したアルゴリズム識別子と，共通鍵ファイルの内容が異なっています。

対処

Web サービスセキュリティ機能定義ファイルを修正して，SecretKeyFile 要素の keytype 属性と共通鍵ファイルの内容が同じになるようにしてください。keytype 属性には，共通鍵生成コマンドの引数で指定したアルゴリズム識別子と同じものを指定する必要があります。

Web サービスセキュリティ機能定義ファイルの SecretKeyFile 要素の keytype 属性に指定する値については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.1 Web サービスセキュリティ機能定義ファイル」を参照してください。共通鍵生成コマンドについては，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「4.1.1　共通鍵生成コマンド（CWSSCreateSecretKey）」を参照してください。

KDCGS0008-E

An error occurred during creation of a secret key. (details = <詳細>)

意味

共通鍵を生成するときにエラーが発生しました。

<詳細>には，エラー内容の詳細が出力されます。

要因

Web サービスセキュリティ機能定義ファイルの SecretKeyFile 要素に指定した共通鍵ファイルから共通鍵を生成するときにエラーが発生しました。次のうちのどちらかがエラーの要因と考えられます。

- SecretKeyFile 要素の指定が間違っている。
- 実行環境の設定が間違っている。

対処

<詳細>の内容に従って，Web サービスセキュリティ機能定義ファイルの設定または実行環境を見直してください。

KDCGS0009-E

The EmbedId attribute value is duplicated.

意味

EmbedId 属性の値が重複しています。

要因

Web サービスセキュリティ機能定義ファイルの EmbedId 属性に指定した値が重複しています。

対処

Web サービスセキュリティ機能定義ファイルを修正して，EmbedId 属性の値が重複しないようにしてください。

Web サービスセキュリティ機能定義ファイルの EmbedId 属性に指定する値については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.1　Web サービスセキュリティ機能定義ファイル」を参照してください。

KDCGS0014-E

An error occurred during reading of a KeyStore. (details = ＜詳細＞)

意味

キーストアの読み込み中にエラーが発生しました。＜詳細＞には，エラーの詳細な内容が出力されます。

要因

Web サービスセキュリティ機能定義ファイルの KeyStore 要素の File 属性に指定したキーストアファイルの読み込みでエラーが発生しました。指定したキーストアファイルが存在しなかったり，ファイルに対するアクセス権がなかったり，ファイルの形式が間違っているおそれがあります。

対処

＜詳細＞の内容に従って，Web サービスセキュリティ機能定義ファイルの KeyStore 要素の File 属性の指定を見直してください。

Web サービスセキュリティ機能定義ファイルで指定する要素および属性については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.1　Web サービスセキュリティ機能定義ファイル」を参照してください。

KDCGS0015-E

A definition is duplicated. (tag name = <要素名>)

意味

要素が重複しています。＜要素名＞には，重複している要素の名称が出力されます。

要因

Web サービスセキュリティ機能定義ファイルの中で，1 回しか指定できない＜要素名＞を複数回指定しています。

対処

Web サービスセキュリティ機能定義ファイルを修正して，＜要素名＞を 1 回だけ指定するようにしてください。

Web サービスセキュリティ機能定義ファイルで指定する要素の指定回数については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.1　Web サービスセキュリティ機能定義ファイル」を参照してください。

19.5　KDCGW で始まるメッセージ

Web サービスセキュリティ機能で出力される KDCGW0001 から KDCGW9999 までのメッセージについて説明します。

KDCGW0002-E

FaultCode：{http://www.hitachi.co.jp/soft/xml/cosminexus/ws/security/0760/faultcode}＜Server.SigningError または Client.SigningError ＞

FaultString：KDCGW0002-E An error occurred during message signature processing. (details ＝ ＜詳細＞)

FaultActor：なし

FaultDetails：なし

意味

メッセージの署名を処理するときにエラーが発生しました。

＜ Server.SigningError または Client.SigningError ＞および<詳細>には，それぞれ次の内容が出力されます。

＜ Server.SigningError または Client.SigningError ＞

エラーがサーバ側で発生したのか，クライアント側で発生したのかを示す文字列が出力されます。サーバ側でエラーが発生している場合は「Server.SigningError」が，クライアント側でエラーが発生している場合は「Client.SigningError」が出力されます。

<詳細>

エラーの詳細が出力されます。

要因

次のうちのどちらかが要因と考えられます。

- Web サービスセキュリティ機能定義ファイルの CanonicalizationMethod 要素，SignatureMethod 要素，または Transform 要素で指定したアルゴリズムが間違っている。
- Web サービスセキュリティ機能定義ファイルの SignatureTarget 要素で指定した署名対象が間違っている。

対処

<詳細>に表示されるエラーの要因を解決してから，再度，処理を実行してください。処理を再度実行する場合は，SOAP アプリケーションまたは Web サービスも再度デプロイする必要があります。

<詳細>に表示されたエラーの要因がわからない場合は，システム管理者に連絡してください。

Web サービスセキュリティ機能定義ファイルの要素で指定する内容については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.1　Web サービスセキュリティ機能定義ファイル」を参照してください。

KDCGW0003-E

FaultCode：{http://www.hitachi.co.jp/soft/xml/cosminexus/ws/security/0760/faultcode}＜Server.EncryptingError または Client.EncryptingError ＞

FaultString：KDCGW0003-E An error occurred during message encryption. (details = ＜詳細＞)

FaultActor：なし

FaultDetails：なし

意味

メッセージの暗号化を処理するときにエラーが発生しました。

＜ Server.EncryptingError または Client.EncryptingError ＞および<詳細>には，それぞれ次の内容が出力されます。

＜ Server.EncryptingError または Client.EncryptingError ＞

エラーがサーバ側で発生したのか，クライアント側で発生したのかを示す文字列が出力されます。サーバ側でエラーが発生している場合は「Server.EncryptingError」が，クライアント側でエラーが発生している場合は「Client.EncryptingError」が出力されます。

<詳細>

エラーの詳細が出力されます。

要因

次のうちのどちらかが要因と考えられます。

- Web サービスセキュリティ機能定義ファイルの ContentsEncryption 要素または KeyEncryption 要素の子要素である EncryptionMethod 要素で指定したアルゴリズムが間違っている。
- Web サービスセキュリティ機能定義ファイルの EncryptionTarget 要素で指定した暗号化対象が間違っている。

対処

<詳細>に表示されるエラーの要因を解決してから，再度，処理を実行してください。処理を再度実行する場合は，SOAP アプリケーションまたは Web サービスも再度デプロイする必要があります。

<詳細>に表示されたエラーの要因がわからない場合は，システム管理者に連絡してください。

Web サービスセキュリティ機能定義ファイルの要素で指定する内容については，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」の「3.1.1　Web サービスセキュリティ機能定義ファイル」を参照してください。

19.6 KEOS02000 から KEOS09999 までのメッセージ

Manager を使用した構築・運用・保守で出力される KEOS02000 から KEOS09999 までのメッセージについて説明します。

KEOS02020-E（C）

共有ライブラリのロードに失敗しました。 ライブラリ名=aa....aa

aa....aa：ライブラリ名

説明

処理を中止します。

対処

次の内容を確認したあと，再度実行してください。

- JavaVM 起動オプションで，java.library.path が指定されているかどうか。
- JavaVM 起動オプションが正しく指定されている場合は，Keymate/Crypto がインストールされているかどうか。

  Keymate/Crypto がインストールされている場合，メモリが不足している可能性があります。処理を実行したホストの管理者に連絡して，メモリ不足を解消してください。

上記以外の場合は，インストールが不完全である可能性があります（必要なファイルがない，または壊れています）。ユーザ管理をアンインストールしたあと，再インストールしてください。

KEOS02102-E（C）

指定された SecretData の暗号化に失敗しました。

説明

指定された SecretData の暗号化に失敗しました。または，暗号鍵ファイルにアクセスできませんでした。

処理を中止します。

対処

次の内容を確認したあと，再度実行してください。

- ユーザ管理のコンフィグレーションファイルに暗号鍵の指定があるか（com.cosminexus.admin.auth.sso.keyfile の指定があるか）。
- 暗号鍵ファイルがあるか。
- 鍵ファイルのアクセス権があるか。

KEOS02152-E（C）

指定された SecretData の復号化に失敗しました。

説明

指定された SecretData の復号化に失敗しました。または，暗号鍵ファイルにアクセスできませんでした。

処理を中止します。

対処

次の内容を確認したあと，再度実行してください。

- ユーザ管理のコンフィグレーションファイルに暗号鍵の指定があるか（com.cosminexus.admin.auth.sso.keyfile の指定があるか）。
- 暗号鍵ファイルがあるか。
- 鍵ファイルのアクセス権があるか。
- 登録時に使用した暗号鍵ファイルと異なる暗号鍵ファイルを指定していないか。

KEOS02202-E（C）

暗号鍵ファイルにアクセスできません。

説明

ユーザ管理のコンフィグレーションファイル内に暗号鍵ファイル名の指定がされていません。または，ディレクトリに書き込み権限がなく暗号鍵ファイルを作成できません。

処理を中止します。

対処

ユーザ管理のコンフィグレーションファイル内の com.cosminexus.admin.auth.sso.keyfile を確認したあと，再度実行してください。

- 暗号鍵ファイル名が指定されているか。
- ファイル名が指定されている場合，ディレクトリのアクセス権はあるか。

KEOS02300-E（C／F）

パスワードの復号化に失敗しました。詳細情報=aa....aa

aa....aa：詳細情報

説明

パスワードの復号化に失敗しました。復号化前の文字列をパスワードとして使用します。

デフォルトの設定で処理を継続します。

対処

パスワードがスクランブル化されているかどうか確認してください。

KEOS13105-E（W／F）

暗号鍵ファイルの作成に失敗しました。詳細情報=aa....aa

aa....aa：例外コード

説明

暗号鍵ファイルを設定情報に適用できませんでした。

［戻る］リンクを表示します。［戻る］リンクを選択すると，暗号鍵ファイルの設定画面に戻ります。

対処

指定ディレクトリに暗号鍵ファイルが存在することを確認してください。また，格納先ディレクトリと暗号鍵ファイルに読み込み権限があることを確認してください。

KEOS13106-E（W／F）

共有ライブラリのロードに失敗しました。詳細情報=aa....aa

aa....aa：例外コード

説明

共有ライブラリのロードに失敗しました。

［戻る］リンクを表示します。

［戻る］リンクを選択すると，呼び出し元画面に戻ります。

対処

Keymate/Crypto がインストールされているかどうかを確認してください。Keymate/Crypto がインストールされている場合，メモリが不足している可能性があります。処理を実行したホストの管理者に連絡して，メモリ不足を解消してください。

それ以外の場合は，インストールに必要なファイルが存在しない，またはファイルが壊れている可能性があります。製品をアンインストールしたあと，再度インストールしてください。

KEOS13107-E（W／F）

secretData の暗号化に失敗しました。詳細情報=aa....aa

aa....aa：例外コード

説明

secretData の暗号化に失敗しました。

［戻る］リンクを表示します。

［戻る］リンクを選択すると，呼び出し元画面に戻ります。

対処

暗号化ライブラリがインストールされていることを確認してください。

KEOS13119-I（W／F）

暗号鍵ファイルの設定を適用しました。

説明

暗号鍵ファイルが作成または適用されました。

［戻る］リンクを表示します。

対処

［戻る］リンクを選択してください。［戻る］リンクを選択すると，暗号鍵ファイルの設定画面に戻ります。

KEOS13125-E（W／F）

入力情報に不正な文字が含まれています。入力情報=aa....aa

aa....aa：入力情報

説明

入力情報が不正です。

［戻る］リンクを表示します。

［戻る］リンクを選択すると，呼び出し元画面に戻ります。

対処

次の内容に従って，入力情報を確認してください。

- レルム名の場合は，英字（A～Z，a～z）および数字（0～9）の文字列であることを確認してください。また，予約されている文字列「mappings」でないことを確認してください。
- ユーザ ID の場合は，英字（A～Z，a～z）および数字（0～9）の文字列であることを確認してください。
- パスワードまたは secretData の場合は，英字（A～Z，a～z），数字（0～9），および特殊文字の文字列であることを確認してください。特殊文字は次に示す記号を意味します。

  （空白） | ! | " | # | $ | % | & | ' | (|) | * | + | , | - | . | / | : | ; | < | = | > | ? | @ | [| ¥ |] | ^ | _（アンダースコア） | ` | { | } | ¦（垂直バー） | ~

KEOS13126-E（W／F）

secretData が違います。

説明

入力された secretData と再入力した secretData が一致しません。

［戻る］リンクを表示します。

［戻る］リンクを選択すると，呼び出し元画面に戻ります。

対処

secretData をもう一度入力してください。

19.7 KEXS で始まるメッセージ

XML Security - Core が出力する KEXS10001 から KEXS99999 までのメッセージについて説明します。

KEXS20006-E

JCE プロバイダが見つかりません。プロバイダ名 = {0}

(要因)

Java セキュリティプロバイダの設定に誤りがあります。

{0}：プロバイダ名

(処理)

例外を通知します。

(対処)

java.security ファイルの設定を確認してください。

KEXS20007-E

JCE アルゴリズムが見つかりません。アルゴリズム名 = {0}, プロバイダ名 = {1}

(要因)

Java セキュリティプロバイダの設定に誤りがあります。

{0}：アルゴリズム名

{1}：プロバイダ名

(処理)

例外を通知します。

(対処)

java.security ファイルの設定を確認してください。

KEXS20008-E

DSA の不正な ASN.1 フォーマットです。

(要因)

DSA の ASN.1 フォーマットが不正です。

(処理)

例外を通知します。

(対処)

java.security ファイルの設定を確認してください。

KEXS20009-E

無効な実装クラスが指定されました。クラス名 = {0}

(要因)

無効な実装クラスが指定されました。

{0}：クラス名

(処理)

例外を通知します。

(対処)

有効な実装クラスを指定してください。

KEXS30010-E

親ノードが存在しません。

(要因)

親ノードを持たない EncryptedData を復号化モードで置換しようとしました。

(処理)

例外を通知します。

(対処)

親ノードを持つ EncryptedData を指定してください。

KEXS30011-E

不正な DOMFragment です。

(要因)

DOMFragment の各ノードに親が存在しない，または DOMFragment の各ノードが兄弟関係ではありません。

(処理)

例外を通知します。

(対処)

各ノードが親を持ち，兄弟関係となる DOMFragment を指定してください。

KEXS40001-E

不正なモードです。

(要因)

実行しようとした処理に対して不正なモードが設定されています。

(処理)

例外を通知します。

(対処)

適切なモードを設定してください。

KEXS40002-E

アルゴリズムが見つかりません。カテゴリ = {0}, アルゴリズム識別子 = {1}

(要因)

無効なアルゴリズムが指定されました。

{0}：カテゴリ

{1}：アルゴリズム識別子

(処理)

例外を通知します。

(対処)

有効なアルゴリズムを指定してください。

KEXS40003-E

鍵リゾルバが設定されていません。

(要因)

鍵リゾルバが設定されていません。

(処理)

例外を通知します。

(対処)

鍵リゾルバを設定してください。

KEXS40004-E

無効なアルゴリズムパラメタが指定されました。

(要因)

無効なアルゴリズムパラメタが指定されました。

(処理)

例外を通知します。

(対処)

有効なアルゴリズムパラメタを指定してください。

KEXS40009-E

鍵解決に失敗しました。

(要因)

鍵解決で鍵を取得できませんでした。

(処理)

例外を通知します。

(対処)

鍵を取得できる，適切な鍵リゾルバを設定してください。

KEXS40010-E

不正なオブジェクトが含まれています。

(要因)

List や Map に不正なオブジェクトが含まれています。

(処理)

例外を通知します。

(対処)

不正なオブジェクトを含まないようにしてください。

KEXS40011-E

enveloped-signature 変換または XPath 関数 here()の使用方法に誤りがあります。

(要因)

enveloped-signature 変換または XPath 関数 here()にノードが結びついていません。

(処理)

例外を通知します。

(対処)

enveloped-signature 変換または XPath 関数 here()の使用方法が正しいか確認してください。

KEXS40012-E

出力先が設定されていません。

(要因)

XMLSerializer の出力先が設定されていません。

(処理)

例外を通知します。

(対処)

出力先を設定してください。

KEXS40013-E

結果は well-formed ではありません。モード = {0}

(要因)

次のような結果になる置換を実行しようとしました。

- 文書要素が二つになる。
- DocumentType ノードの前に要素がある。
- Document ノードの子ノードにコメント，PI 以外のノードが含まれる。

{0}：モード

(処理)

例外を通知します。

(対処)

結果が well-formed になるようにしてください。

KEXS40014-E

無効な鍵長の鍵が指定されました。鍵長 = {0}

(要因)

Key オブジェクトの鍵長が無効です。

{0}：鍵長

(処理)

例外を通知します。

(対処)

有効な鍵長の鍵を指定してください。

KEXS40015-E

無効な鍵合意コンテキストが指定されました。

(要因)

鍵合意コンテキストが無効です。

(処理)

例外を通知します。

(対処)

有効な鍵合意コンテキストを指定してください。

KEXS40016-E

不正な位置のノードが指定されました。モード = {0}

(要因)

暗号化時に，EncryptedData をその子孫にある DOMFragment で置換しようとしました。または，復号化時に，DOMFragment をその子孫または兄弟関係にある EncryptedData と置換しようとしました。

{0}：モード

(処理)

例外を通知します。

(対処)

適切な位置にある EncryptedData または DOMFragment を指定してください。

KEXS50001-E

無効な要素が指定されました。要求要素 = {0}, 指定要素 = {1}

(要因)

無効な要素が指定されました。名前空間 URI またはローカル名に誤りがあります。

{0}：要求要素

{1}：指定要素

(処理)

例外を通知します。

(対処)

有効な要素を指定してください。

KEXS50002-E

KeyInfo コンテントを作成できません。名前空間 URI = {0}, ローカル名 = {1}

(要因)

無効な KeyInfo コンテントが指定されました。

{0}：名前空間 URI

{1}：ローカル名

(処理)

例外を通知します。

(対処)

有効な KeyInfo コンテントを指定してください。

KEXS50003-E

アルゴリズムパラメタが設定されていません。アルゴリズム識別子 = {0}

(要因)

アルゴリズムパラメタが設定されていません。

{0}：アルゴリズム識別子

(処理)

例外を通知します。

(対処)

アルゴリズムパラメタを設定してください。

KEXS50014-E

JCE アルゴリズムの処理に失敗しました。クラス名 = {0}, メソッド名 = {1}, プロバイダ名 = {2}

(要因)

JCE アルゴリズムの処理に失敗しました。または KeyInfo コンテントに不正な値が設定されている可能性があります。

{0}：クラス名

{1}：メソッド名

{2}：プロバイダ名

(処理)

例外を通知します。

(対処)

java.security ファイルの設定を確認してください。

KEXS50015-E

整数フォーマットエラーです。テキスト = {0}

(要因)

整数として認識できないテキストが設定されています。

{0}：テキスト

(処理)

例外を通知します。

(対処)

整数として認識できるテキストを設定してください。

KEXS50016-E

DSA の不正な XML 署名フォーマットです。

(要因)

SignatureValue 要素の DSA 署名値が不正な値です。

(処理)

例外を通知します。

(対処)

有効な DSA 署名値を設定してください。

KEXS50017-E

対応する DOM ノードが存在しません。

(要因)

既存の Element ノードから作成されていない EncryptedData に対して，decryptXML メソッドを適用しようとしました。または，復号化モードで置換しようとしました。

(処理)

例外を通知します。

(対処)

XMLEncryptionFactory クラスの newEncryptedData(XMLSecurityContext,Element)メソッドで作成した EncryptedData を使用してください。

KEXS50018-E

KeySize 要素に無効な値が設定されています。KeySize = {0}

(要因)

KeySize 要素に無効な値が設定されています。

{0}：KeySize 要素の値

(処理)

例外を通知します。

(対処)

有効な値を設定してください。

KEXS50019-E

データのサイズが不正です。データサイズ = {0}

(要因)

指定されたアルゴリズムへの入力データが短過ぎます。またはアルゴリズムで指定されたブロック長の倍数になっていません。

{0}：データサイズ

(処理)

例外を通知します。

(対処)

有効な入力データを指定してください。

KEXS50020-E

完全性チェックに失敗しました。アルゴリズム識別子 = {0}

(要因)

入力データまたは鍵が不正です。

{0}：アルゴリズム識別子

(処理)

例外を通知します。

(対処)

有効な入力データまたは鍵を指定してください。

KEXS50021-E

鍵の作成に必要なパラメタが設定されていません。

(要因)

鍵の作成に必要なパラメタが設定されていません。

(処理)

例外を通知します。

(対処)

必要なパラメタを設定してください。

KEXS50022-E

復号化結果が空です。

(要因)

空のデータが暗号化されています。decryptXML メソッドでは処理できません。

(処理)

例外を通知します。

(対処)

有効なデータを含む EncryptedData を指定してください。

KEXS50023-E

パディングが不正です。アルゴリズム識別子 = {0}

(要因)

入力データまたは鍵が不正です。

{0}：アルゴリズム識別子

(処理)

例外を通知します。

(対処)

有効な入力データまたは鍵を指定してください。

19.8 SSL についてのメッセージ

ここでは，HTTP Server の SSL 処理で出力されるメッセージについて説明します。

19.8.1 メッセージの記述形式

この節でのメッセージの記述形式を次に示します。

XXnnnnn

メッセージテキスト

可変値に関する説明

エラーレベル：エラーログに出力するエラーのレベル

説明

メッセージテキストに対する補足説明

対処

ユーザが実施する対処

なお，「可変値に関する説明」，「対処」はメッセージによって記述しないものもあります。

次に，各項目について説明します。

XXnnnnn

メッセージ ID を表します。

nnnnn

メッセージを出力したプログラムで管理するメッセージ番号を表します。それぞれのメッセージには 5 けたの固有の番号が付いています。

メッセージテキスト

SSL 処理で出力されるメッセージテキストを示します。

なお，メッセージテキスト中の可変値（メッセージが出力される状況によって変わる値）は，「xx....xx」（xx は英小文字）の形式で示します。

（例）

aa....aa：ファイル名

bb....bb：アプリケーション名

エラーレベル

LogLevel ディレクティブで指定したエラーレベルが表示されます。

出力されるエラーレベルは次のとおりです。

- emerg レベル
- alert レベル
- crit レベル
- error レベル
- warn レベル
- notice レベル
- info レベル
- レベルなし
- レベル不定

「レベルなし」のメッセージは，レベル設定のないメッセージで，メッセージテキストだけ出力されます。

「レベル不定」のメッセージは，出力されるメッセージによってエラーレベルが異なります。

説明

メッセージが通知された要因やメッセージを出力した構成ソフトウェアの動作など，メッセージに対する補足説明を示します。

対処

ユーザが実施する対処を表します。

19.8.2 注意事項

notice レベルのメッセージは，LogLevel ディレクティブの指定に関係なく出力されます。

HTTP Server 起動時など，レベル指定解析前には，LogLevel ディレクティブの指定に関係なくメッセージが出力される場合があります。

次に示すメッセージは一部を除き，記載していません。

- HTTP Server の起動時に出力される，コンフィグファイルの文法エラーに伴うメッセージ
- HTTP Server の起動後に出力される，エラーレベルが debug のメッセージ
- HTTP Server の起動後に出力される，エラーレベルがないメッセージ

19.8.3 AH で始まるメッセージ

AH01876

```
mod_ssl/aa....aa compiled against Server: bb....bb cc....cc, Library: dd....dd
```

aa....aa：製品バージョン

bb....bb：製品名

cc....cc：製品バージョン

dd....dd：ライブラリ情報

エラーレベル：info

説明

OpenSSL のライブラリ情報および SSL モジュール情報を出力します。

AH01883

Init: Initialized OpenSSL library

エラーレベル：info

説明

OpenSSL ライブラリの初期化をします。

AH01887

Init: Initializing (virtual) servers for SSL

エラーレベル：info

説明

SSL についてサーバの初期化を行います。

AH01895

Unable to configure verify locations for client authentication

エラーレベル：emerg

説明

クライアント認証に使用する CA 証明書を設定できませんでした。Web サーバを起動しません。

対処

SSLCACertificateFile および SSLCACertificatePath ディレクティブの指定を見直してください。

AH01896

Unable to determine list of acceptable CA certificates for client authentication

エラーレベル：emerg

説明

クライアント認証に使用可能な CA 証明書のリストを決定できませんでした。Web サーバを起動しません。

対処

SSLCACertificateFile および SSLCACertificatePath ディレクティブの指定を見直してください。

AH01897

```
Init: Oops, you want to request client authentication, but no CAs are known for verification!?
[Hint: SSLCACertificate*]
```

エラーレベル：warn

説明

クライアント認証の検証に必要な CA が存在しません。Web サーバは起動処理を続行します。

対処

SSLCACertificateFile および SSLCACertificatePath ディレクティブの指定を見直してください。

AH01898

```
Unable to configure permitted SSL ciphers
```

エラーレベル：emerg

説明

許可する SSL 暗号種別を設定できませんでした。Web サーバを起動しません。

対処

SSLCipherSuite ディレクティブの指定を見直してください。

AH01899

```
Host aa....aa:bb....bb: CRL checking has been enabled, but neither SSLCARevocationFile nor
SSLCARevocationPath is configured
```

aa....aa：ホスト名

bb....bb：ポート番号

エラーレベル：emerg

説明

CRL のチェックが有効になっていますが，SSLCARevocationFile ディレクティブが指定されていません。Web サーバを起動しません。

対処

SSLCARevocationCheck および SSLCARevocationFile ディレクティブの指定を見直してください。

AH01901

Host aa....aa:bb....bb: unable to configure X.509 CRL storage for certificate revocation

aa....aa：ホスト名

bb....bb：ポート番号

エラーレベル：emerg

説明

証明書失効用の CRL を設定できませんでした。Web サーバを起動しません。

対処

SSLCARevocationFile ディレクティブの指定を見直してください。

AH01902

Host aa....aa:bb....bb: X.509 CRL storage locations configured, but CRL checking (SSLCARevocationCheck) is not enabled

aa....aa：ホスト名

bb....bb：ポート番号

エラーレベル：warn

説明

CRL の格納場所は設定されていますが，CRL のチェックが有効になっていません。Web サーバは起動処理を続行します。

対処

CRL のチェックを有効にする場合は，SSLCARevocationCheck ディレクティブを見直してください。

AH01909

aa....aa:bb....bb:cc....cc server certificate does NOT include an ID which matches the server name

aa....aa：ホスト名

bb....bb：ポート番号

cc....cc：インデックス番号

エラーレベル：warn

説明

サーバ証明書にサーバ名称と一致する ID が含まれていません。Web サーバは起動処理を続行します。

対処

サーバ証明書とサーバ名称が一致しているか確認してください。

AH01914

```
Configuring server aa....aa:bb....bb for SSL protocol
```

aa....aa：ホスト名

bb....bb：ポート番号

エラーレベル：info

説明

SSL についてサーバの設定を行います。

AH01915

```
Init: (aa....aa:bb....bb) You configured HTTPS(443) on the standard HTTP(80) port!
```

aa....aa：ホスト名

bb....bb：ポート番号

エラーレベル：warn

説明

SSL が有効となっていますが，HTTP プロトコル標準のポート番号 80 が設定されています。Web サーバは起動処理を続行します。

対処

サーバがリクエストを受け付けるポート番号について見直してください。

AH01916

```
Init: (aa....aa:bb....bb) You configured HTTP(80) on the standard HTTPS(443) port!
```

aa....aa：ホスト名

bb....bb：ポート番号

エラーレベル：warn

説明

SSL が無効となっていますが，HTTPS プロトコル標準のポート番号 443 が設定されています。Web サーバは起動処理を続行します。

対処

サーバがリクエストを受け付けるポート番号について見直してください。

AH01962

Unable to create a new SSL connection from the SSL context

エラーレベル：error

説明

SSL コンテキストから SSL コネクションを生成できませんでした。SSL による接続ができません。

対処

同時に出力されるエラーメッセージを参照してください。

AH01963

Unable to set session id context to 'aa....aa'

aa....aa：サーバを識別するデータ

エラーレベル：error

説明

セション ID コンテキストにデータをセットできませんでした。SSL による接続ができません。

対処

同時に出力されるエラーメッセージを参照してください。

AH01996

SSL handshake failed: HTTP spoken on HTTPS port; trying to send HTML error page

エラーレベル：info

説明

SSL ハンドシェイクに失敗しました：HTTPS を受け付けるポートに HTTP のリクエストを受信しました。

AH01998

Connection closed to child aa....aa with abortive shutdown (server bb....bb:cc....cc)

aa....aa：サーバを識別するユニークな ID

bb....bb：ホスト名

cc....cc：ポート番号

エラーレベル：info

説明

ID で示すサーバとのコネクションが切断されました。

AH02011

```
No acceptable peer certificate available
```

エラーレベル：info

説明

アクセス可能なクライアント証明書がありません。

AH02036

```
Faking HTTP Basic Auth header: "Authorization: aa....aa"
```

aa....aa：ヘッダ値

エラーレベル：info

説明

クライアント証明書による Basic 認証を実行します。

AH02039

```
Certificate Verification: Error (aa....aa): bb....bb
```

aa....aa：エラー番号

bb....bb：エラー文字列

エラーレベル：error

説明

クライアント認証に失敗しました。SSL リクエスト処理を終了します。

AH02040

Certificate Verification: Certificate Chain too long (chain has aa....aa certificates, but maximum allowed are only bb....bb)

aa....aa：CA のチェイン数

bb....bb：SSLVerifyDepth 値

エラーレベル：error

説明

SSLVerifyDepth ディレクティブの指定値より深い階層に位置するクライアント証明書が送られてきたため，検証に失敗しました。SSL リクエスト処理を終了します。

対処

SSLVerifyDepth ディレクティブの指定値を確認してください。クライアント証明書を受け付けないときは対処する必要はありません。

AH02042

rejecting client initiated renegotiation

エラーレベル：error

説明

クライアントからのリネゴシエーション要求を拒否しました。

SSL リクエスト処理を終了します。

AH02219

access to aa....aa failed, reason: SSL connection required

aa....aa：要求されたファイル

エラーレベル：error

説明

SSLRequireSSL ディレクティブを指定したパスに対して，SSL を使用しないでアクセスしました。ステータスコード「403 Forbidden」をクライアントに返し，リクエスト処理を中断します。

AH02231

No SSL protocols available [hint: SSLProtocol]

エラーレベル：emerg

説明

SSLProtocol ディレクティブに利用可能なプロトコルが指定されていません。Web サーバを起動しません。

対処

SSLProtocol ディレクティブの指定を見直してください。

AH02311

```
Fatal error initialising mod_ssl, exiting. See aa....aa for more information
```

aa....aa：エラーログファイル名

エラーレベル：emerg

説明

mod_ssl の初期化中に致命的なエラーが発生しました。Web サーバを起動しません。

対処

メッセージで示すエラーログファイルを参照して，このメッセージのほかに出力されているメッセージのエラーの原因について見直してください。

AH02312

```
Fatal error initialising mod_ssl, exiting.
```

エラーレベル：emerg

説明

mod_ssl の初期化中に致命的なエラーが発生しました。Web サーバを起動しません。

対処

このメッセージのほかに出力されているメッセージのエラーの原因について見直してください。

AH02562

```
Failed to configure certificate aa....aa:bb....bb:cc....cc (with chain), check dd....dd
```

aa....aa：ホスト名

bb....bb：ポート番号

cc....cc：インデックス番号

dd....dd：証明書のファイル名

エラーレベル：emerg

説明

サーバ証明書の設定に失敗しました。Web サーバを起動しません。

対処

SSLCertificateFile ディレクティブで指定した証明書のファイルを見直してください。

AH02564

Failed to configure encrypted (?) private key aa....aa:bb....bb:cc....cc, check dd....dd

aa....aa：ホスト名

bb....bb：ポート番号

cc....cc：インデックス番号

dd....dd：秘密鍵のファイル名

エラーレベル：emerg

説明

サーバ秘密鍵の設定に失敗しました。Web サーバを起動しません。

対処

SSLCertificateKeyFile ディレクティブで指定したサーバ秘密鍵のファイルを見直してください。

AH02565

Certificate and private key aa....aa:bb....bb:cc....cc from dd....dd and ee....ee do not match

aa....aa：ホスト名

bb....bb：ポート番号

cc....cc：インデックス番号

dd....dd：証明書のファイル名

ee....ee：秘密鍵のファイル名

エラーレベル：emerg

説明

サーバ証明書とサーバ秘密鍵の対応が不正です。Web サーバを起動しません。

対処

秘密鍵と証明書が正しいペアで設定されているかどうか SSLCertificateFile ディレクティブおよび SSLCertificateKeyFile ディレクティブを確認してください。

AH02568

Certificate and private key aa....aa:bb....bb:cc....cc configured from dd....dd and ee....ee

aa....aa：ホスト名

bb....bb：ポート番号

cc....cc：インデックス番号

dd....dd：証明書のファイル名

ee....ee：秘密鍵のファイル名

エラーレベル：info

説明

サーバ証明書とサーバ秘密鍵を設定しました。

AH02569

Illegal attempt to re-initialise SSL for server (SSLEngine On should go in the VirtualHost, not in global scope.)

エラーレベル：emerg

説明

サーバに対して不当に SSL の再初期化を実行しようとしました。Web サーバを起動しません。

対処

バーチャルホスト内の SSL の設定を確認してください（バーチャルホスト外で SSL の設定が有効である場合，バーチャルホスト内で一つ以上の SSL 関連ディレクティブを設定しなければなりません）。

AH02572

Failed to configure at least one certificate and key for aa....aa:bb....bb

aa....aa：ホスト名

bb....bb：ポート番号

エラーレベル：emerg

説明

サーバ証明書とサーバ秘密鍵が設定されていません。Web サーバを起動しません。

対処

SSL を有効にする場合，少なくとも一つのサーバ証明書とサーバ秘密鍵を設定する必要があります。

AH02576

Attempting to load encrypted (?) private key aa....aa:bb....bb:cc....cc

aa....aa：ホスト名

bb....bb：ポート番号

cc....cc：インデックス番号

エラーレベル：info

説明

パスワード保護されたサーバ秘密鍵を読み込みます。

AH02577

Init: SSLPassPhraseDialog builtin is not supported on Win32 (key file aa....aa)

aa....aa：サーバ秘密鍵ファイル名

エラーレベル：emerg

説明

Windows 版ではパスワード保護された秘密鍵に対する対話式でのパスワード入力はサポートしていません。Web サーバを起動しません。

対処

サーバ秘密鍵のパスワード保護の解除を検討してください。

AH10104

aa....aa:bb....bb, SSLSrvConfigRec shared from cc....cc:dd....dd

aa....aa：ホスト名

bb....bb：ポート番号

cc....cc：ホスト名

dd....dd：ポート番号

エラーレベル：warn

説明

SSL の設定が共有されています。Web サーバは起動処理を続行します。

対処

SSL の設定を共有してよいかどうか，<VirtualHost>の設定を見直してください。

19.8.4　KH で始まるメッセージ

KH00189

```
SSL handshake interrupted by system: client port aa....aa(bb....bb)(cc....cc)(dd....dd):ee....ee
```

aa....aa：ポート番号

bb....bb：SSL ハンドシェイク処理時間

cc....cc：エラーナンバー値

dd....dd：サーバプロセス ID

ee....ee：SSL ハンドシェイク処理の状態

エラーレベル：info

説明

SSL ハンドシェイク処理が正しく終了しませんでした。
SSL リクエスト処理を終了します。

KH00190

```
SSL handshake interrupted by system: client port aa....aa
```

aa....aa：ポート番号

エラーレベル：info

説明

SSL ハンドシェイク処理が正しく終了しませんでした。
SSL リクエスト処理を終了します。

KH00435

```
Unsupported protocol is ignored: SSLv3
```

エラーレベル：warn

説明

SSLProtocol ディレクティブに未サポートの SSLv3 プロトコルが指定されました。SSLv3 プロトコルの指定を無視して，起動処理を続行します。

対処

SSLProtocol ディレクティブの指定を見直してください。

KH00436

Certificate Verification: Error (aa....aa): bb....bb [cc....cc]

aa....aa：エラー番号

bb....bb：エラー文字列

cc....cc：証明書情報

エラーレベル：error

説明

クライアント認証に失敗しました。SSL リクエスト処理を終了します。

KH00439

SSL Library Error: aa....aa

aa....aa：詳細情報

エラーレベル：不定

説明

SSL ライブラリでエラーが発生しました。

Web サーバ起動中の場合は，起動処理を中断します。SSL リクエスト処理中の場合は，SSL リクエスト処理を中断します。

対処

このメッセージで出力される詳細情報について見直してください。

KH00440

SSL library error aa....aa in handshake (server bb....bb:cc....cc) (dd....dd)(ee....ee)(ff....ff):gg....gg

aa....aa：エラー番号

bb....bb：ホスト名

cc....cc：ポート番号

dd....dd：SSL ハンドシェイク処理時間

ee....ee：エラーナンバー値

ff....ff：サーバプロセス ID

gg....gg：SSL ハンドシェイク処理の状態

エラーレベル：error

説明

SSL ハンドシェイク処理が正しく終了しませんでした。
SSL リクエスト処理を終了します。

KH00441

```
SSL library error aa....aa in handshake (server bb....bb:cc....cc)
```

aa....aa：エラー番号

bb....bb：ホスト名

cc....cc：ポート番号

エラーレベル：error

説明

SSL ハンドシェイク処理が正しく終了しませんでした。
SSL リクエスト処理を終了します。

KH00442

```
access to aa....aa failed, reason: Cipher bb....bb is forbidden
```

aa....aa：ファイル名

bb....bb：暗号種別

エラーレベル：error

説明

SSLBanCipher ディレクティブで指定した暗号種別を使用してアクセスしました。
ステータスコード「403 Forbidden」を返してリクエスト処理を終了します。

KH00443

```
ap_os_proc_filepath() failed.
```

エラーレベル：error

説明

起動コマンドの絶対パス取得に失敗しました。
Web サーバは起動処理を中断します。

KH00444

apr_filepath_merge() failed.

エラーレベル：error

説明

SSL ライブラリのパス生成に失敗しました。
Web サーバは起動処理を中断します。

KH00445

Could not load the ssl library aa..aa

aa....aa：ファイル名

エラーレベル：error

説明

SSL ライブラリのロードに失敗しました。
Web サーバは起動処理を中断します。

付録

付録 A　各バージョンでの主な機能変更

ここでは，11-20 よりも前のアプリケーションサーバの各バージョンでの主な機能の変更について，変更目的ごとに説明します。11-20 での主な機能変更については，「1.4　アプリケーションサーバ 11-20 での主な機能変更」を参照してください。

説明内容は次のとおりです。

- アプリケーションサーバの各バージョンで変更になった主な機能と，その概要を説明しています。機能の詳細については，「参照先マニュアル」の「参照個所」の記述を確認してください。「参照先マニュアル」および「参照個所」には，その機能についての 11-20 のマニュアルでの主な記載個所を記載しています。
- 「参照先マニュアル」に示したマニュアル名の「アプリケーションサーバ」は省略しています。

付録 A.1　11-10 での主な機能変更

(1)　導入・構築の容易性強化

導入・構築の容易性強化を目的として変更した項目を次の表に示します。

表 A–1　導入・構築の容易性強化を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
ライブラリ競合回避機能の追加	クラス・リソースをロードするときの検索順序を変更し，ユーザアプリケーションに含まれるライブラリを優先して参照できるようにする機能を追加しました。	機能解説 基本・開発編 (コンテナ共通機能)	付録 B.4

(2)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–2　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
JSP2.3 への対応	JSP2.3 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	8 章
コンテナ管理の EntityManager への対応	JPA 2.1 に対応したコンテナ管理の EntityManager に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	5 章
Interceptors 1.2 への対応	Interceptors 1.2 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	15 章

(3)　信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A–3　信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
セッションマネージャの指定機能の追加	クラウド環境利用時に HTTP セッションのセッションフェイルオーバを利用できるようにする機能を追加しました。	機能解説 基本・開発編 (Web コンテナ)	2.22

付録 A.2　11-00 での主な機能変更

(1)　導入・構築の容易性強化

導入・構築の容易性強化を目的として変更した項目を次の表に示します。

表 A–4　導入・構築の容易性強化を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
開発環境の Windows Server 対応	クラウド環境上にアプリケーション開発環境を構築できるよう，uCosminexus Developer のサポート OS に Windows Server OS を追加しました。	－	－

(2)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–5　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
Servlet 3.0/3.1 への対応	Servlet 3.0 の非同期サーブレット，および Servlet 3.1 の非同期 I/O 系 API に対応しました。	機能解説 基本・開発編 (Web コンテナ)	8.1
EL 3.0 への対応	EL 3.0 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	2.3.3
JSF 2.2 への対応	JSF 2.2 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	3 章
JAX-RS 2,0 への対応	JAX-RS 2.0 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	4 章
WebSocket 1.0 への対応	WebSocket 1.0 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	5 章

項目	変更の概要	参照先マニュアル	参照個所
NIO HTTP サーバ機能の追加	従来のリダイレクタ機能やインプロセス HTTP サーバ機能に代わり，非同期サーブレットや WebSocket などのノンブロッキング I/O 処理に対応したインプロセスの HTTP サーバとして，NIO HTTP サーバ機能を追加しました。	機能解説 基本・開発編 (Web コンテナ)	7 章
JPA 2.1 への対応	JPA 2.1 に対応し，JPA 2.1 対応の JPA プロバイダを利用できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	6 章
CDI 1.2 への対応	CDI 1.2 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	9 章
BV 1.1 への対応	Bean Validation 1.1 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	10 章
Java Batch 1.0 への対応	Batch Applications for the Java Platform （Java Batch） 1.0 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	11 章
JSON-P 1.0 への対応	Java API for JSON Processing（JSON-P） 1.0 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	12 章
Concurrency Utilities 1.0 への対応	Concurrency Utilities for Java EE 1.0 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	14 章
WebSocket 通信への対応	HTTP Server から J2EE サーバに WebSocket 通信を中継する機能を追加しました。	HTTP Server	4.15

(3) 信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A-6 信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
暗号化通信モジュールの変更	HTTP Server の暗号化通信モジュールとして mod_ssl を採用しました。	HTTP Server	5 章，付録 H

(4) そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A-7 そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
V9 互換モードの追加	Version 9 以前の J2EE サーバから移行するユーザ向けに，Version 9 との互換性を維持するための V9 互換モードを追加しました。	機能解説 保守／移行編	10.3.4

付録 A.3　09-87 での主な機能変更

(1)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–8　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
Java SE 11 への対応	Java SE 11 の機能が使用できるようになりました。	機能解説 保守／移行編	9 章

付録 A.4　09-80 での主な機能変更

(1)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–9　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
JAX-RS 機能におけるラムダ式の使用	web.xml のサーブレット初期化パラメタに指定したパッケージとそのサブパッケージに含まれるクラスで，ラムダ式が使用できるようになりました。	Web サービス開発ガイド	11.2
Java SE 9 への対応	Java SE 9 の機能が使用できるようになりました。	機能解説 保守／移行編	9 章

(2)　信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A–10　信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
Web サーバの Apache2.4 のサポート	Web サーバのベースバージョンとして Apache2.4 をサポートしました。	HTTP Server	6 章，付録 G
SSL 通信での楕円曲線暗号の利用	楕円曲線暗号を利用した SSL 通信ができるようになりました。	HTTP Server	5 章，付録 G
SSL ライブラリの変更	SSL 機能を提供する SSL ライブラリを OpenSSL に変更しました。	HTTP Server	5 章，付録 G

付録 A.5　09-70 での主な機能変更

(1)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–11　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
運用管理ポータルでの JSP コンパイルバージョンの追加	J2EE サーバでの JSP から生成されたサーブレットのコンパイル方法に「JDK1.7 の仕様に従ったコンパイル」と「JDK7 の仕様に従ったコンパイル」を追加しました。	運用管理ポータル操作ガイド	10.8.4
		リファレンス 定義編(サーバ定義)	4.11.2
JDK8 でのメタスペース対応	JavaVM の起動で使用している Permanent 領域用のオプションを Metaspace 領域用のオプションに変更しました。	システム構築・運用ガイド	付録 A.2
		運用管理ポータル操作ガイド	10.8.7
		リファレンス 定義編(サーバ定義)	5.2.1, 5.2.2, 8.2.3
統合ユーザ管理でのユーザ認証の SHA-2 対応	統合ユーザ管理でのユーザ認証のハッシュアルゴリズムとして SHA-224，SHA-256，SHA-384，SHA-512 を追加しました。	このマニュアル	5.3.1, 5.3.9, 5.10.7, 11.4.3, 12.4.3, 12.5.3, 13.2, 14.2.2
Red Hat Enterprise Linux Server 7 での自動起動と自動再起動および自動停止の追加	Red Hat Enterprise Linux Server 7 での Management Server と運用管理エージェントの自動起動と自動再起動および自動停止方法を追加する。	機能解説 運用／監視／連携編	2.6.3, 2.6.4, 2.6.5
		リファレンス コマンド編	7.2

(2)　運用性の維持・向上

運用性の維持・向上を目的として変更した項目を次の表に示します。

表 A–12　運用性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
V9.7 へのバージョンアップ対応	バージョンアップ時の JavaVM の起動で使用している Permanent 領域用のオプションを Metaspace 領域用のオプションに変更する手順を追加しました。	機能解説 保守／移行編	10.3.1, 10.3.2, 10.3.5

項目	変更の概要	参照先マニュアル	参照個所
WAR による運用	WAR ファイルだけで構成された WAR アプリケーションを J2EE サーバにデプロイできるようになりました。	機能解説 基本・開発編 (Web コンテナ)	2.2.1
		機能解説 基本・開発編 (コンテナ共通機能)	18.9
		リファレンス コマンド編	cjimportwar (WAR アプリケーションのインポート)
運用管理機能の同期実行による起動と停止	運用管理機能（Management Server および運用管理エージェント）の起動および停止を，同期実行するオプションを追加しました。	機能解説 運用／監視／連携編	2.6.1, 2.6.2, 2.6.3, 2.6.4
		リファレンス コマンド編	adminagentctl (運用管理エージェントの起動と停止), mngautorun（自動起動および自動再起動の設定／設定解除), mngsvrctl (Management Server の起動／停止／セットアップ)
明示管理ヒープ機能での Explicit メモリブロックの強制解放	javagc コマンドで，Explicit メモリブロックの解放処理を任意のタイミングで実行できるようになりました。	機能解説 拡張編	7.6.1, 7.9
		リファレンス コマンド編	javagc (GC の強制発生)

(3) そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A-13 そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
snapshot ログの収集対象	snapshot ログの収集対象として JavaVM イベントログと Management Server のスレッドダンプを追加しました。	機能解説 保守／移行編	付録 A.2
cjenvsetup コマンドのログ出力	Component Container 管理者のセットアップ（cjenvsetup コマンド）の実行情報がメッセージログに出力されるようになりました。	システム構築・運用ガイド	4.1.4
		機能解説 保守／移行編	4.20
		リファレンス コマンド編	cjenvsetup（Component Container 管理者のセットアップ）
BIG-IP v11 のサポート	使用できる負荷分散機の種類に BIG-IP v11 が追加になりました。	システム構築・運用ガイド	4.7.2
		仮想化システム構築・運用ガイド	2.1
明示管理ヒープ機能のイベントログへの CPU 時間の出力	Explicit メモリブロック解放処理に掛かった CPU 時間が，明示管理ヒープ機能のイベントログに出力されるようになりました。	機能解説 保守／移行編	5.11.3
ユーザ拡張性能解析トレースの機能拡張	ユーザ拡張性能解析トレースで，次の機能を追加しました。 • トレース対象の指定方法を通常のメソッド単位の指定方法に加えて，パッケージ単位またはクラス単位で指定できるようになりました。 • 使用できるイベント ID の範囲を拡張しました。 • ユーザ拡張性能解析トレース設定ファイルに指定できる行数の制限を緩和しました。 • ユーザ拡張性能解析トレース設定ファイルでトレース取得レベルを指定できるようになりました。	機能解説 保守／移行編	7.5.2, 7.5.3, 8.25.1
Session Bean の非同期呼び出し使用時の情報解析向上	PRF トレースのルートアプリケーション情報を使用して，呼び出し元と呼び出し先のリクエストを突き合わせることができるようになりました。	機能解説 基本・開発編 (EJB コンテナ)	2.17.3

付録 A.6　09-60 での主な機能変更

(1)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–14　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
G1GC への対応	G1GC を選択できるようになりました。	システム設計ガイド	7.15
		リファレンス 定義編(サーバ定義)	14.5
圧縮オブジェクトポインタ機能への対応	圧縮オブジェクトポインタ機能を使用できるようになりました。	機能解説 保守/移行編	9.16

(2)　信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A–15　信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
ファイナライズ滞留解消機能の追加	ファイナライズ処理の滞留を解消でき，OS 資源の解放遅れなどの発生を抑止できるようになりました。	機能解説 保守/移行編	9.17

(3)　そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A–16　そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
ログファイルの非同期出力機能の追加	ログのファイル出力を非同期でできるようになりました。	リファレンス 定義編(サーバ定義)	14.2

付録 A.7　09-50 での主な機能変更

(1)　開発生産性の向上

開発生産性の向上を目的として変更した項目を次の表に示します。

表 A-17　開発生産性の向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
Eclipse セットアップの簡略化	GUI を利用して Eclipse 環境をセットアップできるようになりました。	アプリケーション開発ガイド	1.1.5, 2.4
ユーザ拡張性能解析トレースを使ったデバッグ支援	ユーザ拡張性能解析トレース設定ファイルを開発環境で作成できるようになりました。	アプリケーション開発ガイド	1.1.3, 6.4

(2)　導入・構築の容易性強化

導入・構築の容易性強化を目的として変更した項目を次の表に示します。

表 A-18　導入・構築の容易性強化を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
仮想化環境でのシステム構成パターンの拡充	仮想化環境で使用できるティアの種類（http-tier，j2ee-tier および ctm-tier）が増えました。これによって，次のシステム構成パターンが構築できるようになりました。 • Web サーバと J2EE サーバを別のホストに配置するパターン • フロントエンド（サーブレット，JSP）とバックエンド（EJB）を分けて配置するパターン • CTM を使用するパターン	仮想化システム構築・運用ガイド	1.1.2

(3)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A-19　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
JDBC 4.0 仕様への対応	DB Connector で JDBC 4.0 仕様の HiRDB Type4 JDBC Driver，および SQL Server の JDBC ドライバに対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	3.6.3
Portable Global JNDI 名での命名規則の緩和	Portable Global JNDI 名に使用できる文字を追加しました。	機能解説 基本・開発編 (コンテナ共通機能)	2.4.3
Servlet 3.0 仕様への対応	Servlet 3.0 の HTTP Cookie の名称，および URL のパスパラメタ名の変更が，Servlet 2.5 以前のバージョンでも使用できるようになりました。	機能解説 基本・開発編 (Web コンテナ)	2.7
Bean Validation と連携できるアプリケーションの適用拡大	CDI やユーザアプリケーションでも Bean Validation を使って検証できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	10 章
JavaMail への対応	JavaMail 1.4 に準拠した API を使用したメール送受信機能を利用できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	8 章

項目	変更の概要	参照先マニュアル	参照個所
javacore コマンドが使用できる OS の適用拡大	javacore コマンドを使って，Windows のスレッドダンプを取得できるようになりました。	リファレンス コマンド編	javacore（スレッドダンプの取得／Windows の場合）

(4) 信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A-20 信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
コードキャッシュ領域の枯渇回避	システムで使用しているコードキャッシュ領域のサイズを確認して，領域が枯渇する前にしきい値を変更して領域枯渇するのを回避できるようになりました。	システム設計ガイド	7.2.6
		機能解説 保守／移行編	5.7.2, 5.7.3
		リファレンス 定義編(サーバ定義)	14.1, 14.2, 14.4
明示管理ヒープ機能の効率的な適用への対応	自動解放処理時間を短縮し，明示管理ヒープ機能を効率的に適用するための機能として，Explicit ヒープに移動するオブジェクトを制御できる次の機能を追加しました。 • Explicit メモリブロックへのオブジェクト移動制御機能 • 明示管理ヒープ機能適用除外クラス指定機能 • Explicit ヒープ情報へのオブジェクト解放率情報の出力	システム設計ガイド	7.14.6
		機能解説 拡張編	7.2.2, 7.6.5, 7.10, 7.13.1, 7.13.3
		機能解説 保守／移行編	5.5
クラス別統計情報の出力範囲拡大	クラス別統計情報を含んだ拡張スレッドダンプに，static フィールドを基点とした参照関係を出力できるようになりました。	機能解説 保守／移行編	9.6

(5) 運用性の維持・向上

運用性の維持・向上を目的として変更した項目を次の表に示します。

表 A-21 運用性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
EADs セッションフェイルオーバ機能のサポート	EADs と連携してセッションフェイルオーバ機能を実現する EADs セッションフェイルオーバ機能をサポートしました。	機能解説 拡張編	5 章

項目	変更の概要	参照先マニュアル	参照個所
WAR による運用	WAR ファイルだけで構成された WAR アプリケーションを J2EE サーバにデプロイできるようになりました。	機能解説 基本・開発編 (Web コンテナ)	2.2.1
		機能解説 基本・開発編 (コンテナ共通機能)	18.9
		リファレンス コマンド編	cjimport war (WAR アプリケーションのインポート)
運用管理機能の同期実行による起動と停止	運用管理機能（Management Server および運用管理エージェント）の起動および停止を，同期実行するオプションを追加しました。	機能解説 運用／監視／連携編	2.6.1, 2.6.2, 2.6.3, 2.6.4
		リファレンス コマンド編	adminagentctl (運用管理エージェントの起動と停止), mngautorun（自動起動および自動再起動の設定／設定解除), mngsvrctl (Management Server の起動／停止／セットアップ)
明示管理ヒープ機能での Explicit メモリブロックの強制解放	javagc コマンドで，Explicit メモリブロックの解放処理を任意のタイミングで実行できるようになりました。	機能解説 拡張編	7.6.1, 7.9
		リファレンス コマンド編	javagc (GC の強制発生)

(6) そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A-22 そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
定義情報の取得	snapshotlog（snapshot ログの収集）コマンドで定義ファイルだけを収集できるようになりました。	機能解説 保守／移行編	2.3
		リファレンス コマンド編	snapshotlog（snapshot ログの収集）
cjenvsetup コマンドのログ出力	Component Container 管理者のセットアップ（cjenvsetup コマンド）の実行情報がメッセージログに出力されるようになりました。	システム構築・運用ガイド	4.1.4
		機能解説 保守／移行編	4.20
		リファレンス コマンド編	cjenvsetup（Component Container 管理者のセットアップ）
BIG-IP v11 のサポート	使用できる負荷分散機の種類に BIG-IP v11 が追加になりました。	システム構築・運用ガイド	4.7.2
		仮想化システム構築・運用ガイド	2.1
明示管理ヒープ機能のイベントログへの CPU 時間の出力	Explicit メモリブロック解放処理に掛かった CPU 時間が，明示管理ヒープ機能のイベントログに出力されるようになりました。	機能解説 保守／移行編	5.11.3
ユーザ拡張性能解析トレースの機能拡張	ユーザ拡張性能解析トレースで，次の機能を追加しました。 • トレース対象の指定方法を通常のメソッド単位の指定方法に加えて，パッケージ単位またはクラス単位で指定できるようになりました。 • 使用できるイベント ID の範囲を拡張しました。 • ユーザ拡張性能解析トレース設定ファイルに指定できる行数の制限を緩和しました。 • ユーザ拡張性能解析トレース設定ファイルでトレース取得レベルを指定できるようになりました。	機能解説 保守／移行編	7.5.2, 7.5.3, 8.25.1
Session Bean の非同期呼び出し使用時の情報解析向上	PRF トレースのルートアプリケーション情報を使用して，呼び出し元と呼び出し先のリクエストを突き合わせることができるようになりました。	機能解説 基本・開発編 (EJB コンテナ)	2.17.3

付録 A.8　09-00 での主な機能変更

(1)　導入・構築の容易性強化

導入・構築の容易性強化を目的として変更した項目を次の表に示します。

表 A–23　導入・構築の容易性強化を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
仮想化環境での構築・運用の操作対象単位の変更	仮想化環境の構築・運用時の操作対象単位が仮想サーバから仮想サーバグループへ変更になりました。仮想サーバグループの情報を定義したファイルを使用して，複数の仮想サーバを管理ユニットへ一括で登録できるようになりました。	仮想化システム構築・運用ガイド	1.1.2
セットアップウィザードによる構築環境の制限解除	セットアップウィザードを使用して構築できる環境の制限が解除されました。ほかの機能で構築した環境があってもアンセットアップされて，セットアップウィザードで構築できるようになりました。	システム構築・運用ガイド	2.2.7
構築環境の削除手順の簡略化	Management Server を使用して構築したシステム環境を削除する機能（mngunsetup コマンド）の追加によって，削除手順を簡略化しました。	システム構築・運用ガイド	4.1.37
		運用管理ポータル操作ガイド	3.6，5.4
		リファレンス コマンド編	mngunsetup（Management Server の構築環境の削除）

(2)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–24　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
Servlet 3.0 への対応	Servlet 3.0 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	8 章
EJB 3.1 への対応	EJB 3.1 に対応しました。	機能解説 基本・開発編 (EJB コンテナ)	2 章
JSF 2.1 への対応	JSF 2.1 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	3 章

項目	変更の概要	参照先マニュアル	参照個所
JSTL 1.2 への対応	JSTL 1.2 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	3 章
CDI 1.0 への対応	CDI 1.0 に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	9 章
Portable Global JNDI 名の利用	Portable Global JNDI 名を利用したオブジェクトのルックアップができるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	2.4
JAX-WS 2.2 への対応	JAX-WS 2.2 に対応しました。	Web サービス開発ガイド	1.1, 16.1.5, 16.1.7, 16.2.1, 16.2.6, 16.2.10, 16.2.12, 16.2.13, 16.2.14, 16.2.16, 16.2.17, 16.2.18, 16.2.20, 16.2.22, 19.1, 19.2.3, 37.2, 37.6.1, 37.6.2, 37.6.3
JAX-RS 1.1 への対応	JAX-RS 1.1 に対応しました。	Web サービス開発ガイド	1.1, 1.2.2, 1.3.2, 1.4.2, 1.5.1, 1.6, 2.3, 11 章, 12 章, 13 章, 17 章, 24 章, 39 章

(3) 信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A-25　信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
SSL/TLS 通信での TLSv1.2 の使用	RSA BSAFE SSL-J を使用して，TLSv1.2 を含むセキュリティ・プロトコルで SSL/TLS 通信ができるようになりました。	このマニュアル	7.3

(4)　運用性の維持・向上

運用性の維持・向上を目的として変更した項目を次の表に示します。

表 A-26　運用性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
Web コンテナ全体の実行待ちキューの総和の監視	Web コンテナ全体の実行待ちキューの総和を稼働情報に出力して監視できるようになりました。	機能解説 運用／監視／連携編	3 章
アプリケーションの性能解析トレース（ユーザ拡張トレース）の出力	ユーザが開発したアプリケーションの処理性能を解析するための性能解析トレースを，アプリケーションの変更をしないで出力できるようになりました。	機能解説 保守／移行編	7 章
仮想化環境でのユーザスクリプトを使用した運用	任意のタイミングでユーザ作成のスクリプト（ユーザスクリプト）を仮想サーバ上で実行できるようになりました。	仮想化システム構築・運用ガイド	7.8
運用管理ポータル改善	運用管理ポータルの次の画面で，手順を示すメッセージを画面に表示するように変更しました。 • ［設定情報の配布］画面 • Web サーバ，J2EE サーバおよび SFO サーバの起動画面 • Web サーバクラスタと J2EE サーバクラスタの一括起動，一括再起動および起動画面	運用管理ポータル操作ガイド	10.10.1, 11.9.2, 11.10.2, 11.10.4, 11.10.6, 11.11.2, 11.12.2, 11.12.4, 11.12.6
運用管理機能の再起動機能の追加	運用管理機能（Management Server および運用管理エージェント）で自動再起動が設定できるようになり，運用管理機能で障害が発生した場合でも運用が継続できるようになりました。また，自動起動の設定方法も変更になりました。	機能解説 運用／監視／連携編	2.4.1, 2.4.2, 2.6.3, 2.6.4
		リファレンス コマンド編	mngautorun（自動起動および自動再起動の設定／設定解除）

(5)　そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A-27　そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
ログ出力時のファイル切り替え単位の変更	ログ出力時に，日付ごとに出力先のファイルを切り替えられるようになりました。	機能解説 保守／移行編	3.2.1
Web サーバの名称の変更	アプリケーションサーバに含まれる Web サーバの名称を HTTP Server に変更しました。	HTTP Server	－
BIG-IP の API（SOAP アーキテクチャ）を使用した直接接続への対応	BIG-IP（負荷分散機）で API（SOAP アーキテクチャ）を使用した直接接続に対応しました。 また，API を使用した直接接続を使用する場合に，負荷分散機の接続環境を設定する方法が変更になりました。	システム構築・運用ガイド	4.7.3，付録 J
		仮想化システム構築・運用ガイド	2.1，付録 C
		このマニュアル	8.2，8.4，8.5，8.6，18.2.1，18.2.2，18.2.3

（凡例）－：マニュアル全体を参照する

付録 A.9　08-70 での主な機能変更

(1)　導入・構築の容易性強化

導入・構築の容易性強化を目的として変更した項目を次の表に示します。

表 A-28　導入・構築の容易性強化を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
運用管理ポータル改善	運用管理ポータルの画面で，リソースアダプタの属性を定義するプロパティ（Connector 属性ファイルの設定内容）の設定，および接続テストができるようになりました。また，運用管理ポータルの画面で，J2EE アプリケーション（ear ファイルおよび zip ファイル）を Management Server にアップロードできるようになりました。	ファーストステップガイド	3.5
		運用管理ポータル操作ガイド	－
page/tag ディレクティブの import 属性暗黙インポート機能の追加	page/tag ディレクティブの import 属性暗黙インポート機能を使用できるようになりました。	機能解説 基本・開発編 (Web コンテナ)	2.3.7
仮想化環境での JP1 製品に対する環境設定の自動化対応	仮想サーバへのアプリケーションサーバ構築時に，仮想サーバに対する JP1 製品の環境設定を，フックスクリプトで自動的に設定できるようになりました。	仮想化システム構築・運用ガイド	7.7.2
統合ユーザ管理機能の改善	ユーザ情報リポジトリでデータベースを使用する場合に，データベース製品の JDBC ドライバを使用して，	このマニュアル	5 章

項目	変更の概要	参照先マニュアル	参照個所
	データベースに接続できるようになりました。 DABroker Library の JDBC ドライバによるデータベース接続はサポート外になりました。 簡易構築定義ファイルおよび運用管理ポータルの画面で，統合ユーザ管理機能に関する設定ができるようになりました。 また，Active Directory の場合，DN で日本語などの 2 バイト文字に対応しました。	運用管理ポータル操作ガイド	3.5, 10.8.1
		このマニュアル	14.2.2
HTTP Server 設定項目の拡充	簡易構築定義ファイルおよび運用管理ポータルの画面で，HTTP Server の動作環境を定義するディレクティブ（httpsd.conf の設定内容）を直接設定できるようになりました。	システム構築・運用ガイド	4.1.21
		運用管理ポータル操作ガイド	10.9.1
		リファレンス 定義編(サーバ定義)	4.10

（凡例）－：マニュアル全体を参照する

(2)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–29　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
ejb-jar.xml の指定項目の追加	ejb-jar.xml に，クラスレベルインターセプタおよびメソッドレベルインターセプタを指定できるようになりました。	機能解説 基本・開発編(EJB コンテナ)	2.15
パラレルコピーガーベージコレクションへの対応	パラレルコピーガーベージコレクションを選択できるようになりました。	リファレンス 定義編(サーバ定義)	14.5
Connector 1.5 仕様に準拠した Inbound リソースアダプタのグローバルトランザクションへの対応	Connector 1.5 仕様に準拠したリソースアダプタで Transacted Delivery を使用できるようにしました。これによって，Message-driven Bean を呼び出す EIS がグローバルトランザクションに参加できるようになりました。	機能解説 基本・開発編(コンテナ共通機能)	3.16.3
TP1 インバウンドアダプタの MHP への対応	TP1 インバウンドアダプタを使用してアプリケーションサーバを呼び出す OpenTP1 のクライアントとして，MHP を使用できるようになりました。	機能解説 基本・開発編(コンテナ共通機能)	4 章
cjrarupdate コマンドの FTP インバウンドアダプタへの対応	cjrarupdate コマンドでバージョンアップできるリソースアダプタに FTP インバウンドアダプタを追加しました。	リファレンス コマンド編	2.2

(3)　信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A-30　信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
データベースセッションフェイルオーバ機能の改善	性能を重視するシステムで，グローバルセッション情報を格納したデータベースのロックを取得しないモードを選択できるようになりました。また，データベースを更新しない，参照専用のリクエストを定義できるようになりました。	機能解説 拡張編	6 章
OutOfMemory ハンドリング機能の対象となる処理の拡大	OutOfMemory ハンドリング機能の対象となる処理を追加しました。	機能解説 保守／移行編	2.5.4
		リファレンス 定義編 (サーバ定義)	14.2
HTTP セッションで利用する Explicit ヒープの省メモリ化機能の追加	HTTP セッションで利用する Explicit ヒープのメモリ使用量を抑止する機能を追加しました。	機能解説 拡張編	7.11

(4)　運用性の維持・向上

運用性の維持・向上を目的として変更した項目を次の表に示します。

表 A-31　運用性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
仮想化環境での JP1 製品を使用したユーザ認証への対応（クラウド運用対応）	JP1 連携時に，JP1 製品の認証サーバを利用して，仮想サーバマネージャを使用するユーザを管理・認証できるようになりました。	仮想化システム構築・運用ガイド	1.2.2，3 章，4 章，5 章，6 章，7.9

(5)　そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A-32　そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
負荷分散機への API（REST アーキテクチャ）を使用した直接接続の対応	負荷分散機への接続方法として，API（REST アーキテクチャ）を使用した直接接続に対応しました。 また，使用できる負荷分散機の種類に ACOS（AX2500）が追加になりました。	システム構築・運用ガイド	4.7.2，4.7.3
		仮想化システム構築・運用ガイド	2.1
		リファレンス 定義編 (サーバ定義)	4.2.4
snapshot ログ収集時のタイムアウトへの対応と収集対象の改善	snapshot ログの収集が指定した時間で終了（タイムアウト）できるようになりました。一次送付資料として収集される内容が変更になりました。	機能解説 保守／移行編	付録 A

付録 A.10　08-53 での主な機能変更

(1)　導入・構築の容易性強化

導入・構築の容易性強化を目的として変更した項目を次の表に示します。

表 A–33　導入・構築の容易性強化を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
さまざまなハイパーバイザに対応した仮想化環境の構築	さまざまなハイパーバイザを使用して実現する仮想サーバ上に，アプリケーションサーバを構築できるようになりました。 また，複数のハイパーバイザが混在する環境にも対応しました。	仮想化システム構築・運用ガイド	2 章，3 章，5 章

(2)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–34　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
トランザクション連携に対応した OpenTP1 からの呼び出し	OpenTP1 からアプリケーションサーバ上で動作する Message-driven Bean を呼び出すときに，トランザクション連携ができるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	4 章
JavaMail	POP3 に準拠したメールサーバと連携して，JavaMail 1.3 に準拠した API を使用したメール受信機能を利用できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	8 章

(3)　信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A–35　信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
JavaVM のトラブルシュート機能強化	JavaVM のトラブルシュート機能として，次の機能が使用できるようになりました。 • OutOfMemoryError 発生時の動作を変更できるようになりました。 • JIT コンパイル時に，C ヒープ確保量の上限値を設定できるようになりました。 • スレッド数の上限値を設定できるようになりました。 • 拡張 verbosegc 情報の出力項目を拡張しました。	機能解説 保守／移行編	4 章，5 章，9 章

(4)　運用性の維持・向上

運用性の維持・向上を目的として変更した項目を次の表に示します。

表 A–36　運用性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
JP1/ITRM への対応	IT リソースを一元管理する製品である JP1/ITRM に対応しました。	仮想化システム構築・運用ガイド	1.3，2.1

(5)　そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A–37　そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
Microsoft IIS 7.0 および Microsoft IIS 7.5 への対応	Web サーバとして Microsoft IIS 7.0 および Microsoft IIS 7.5 に対応しました。	－	－
HiRDB Version 9 および SQL Server 2008 への対応	データベースとして次の製品に対応しました。 • HiRDB Server Version 9 • HiRDB/Developer's Kit Version 9 • HiRDB/Run Time Version 9 • SQL Server 2008 また，SQL Server 2008 に対応する JDBC ドライバとして，SQL Server JDBC Driver に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	3 章

（凡例）－：該当なし。

付録 A.11　08-50 での主な機能変更

(1)　導入・構築の容易性強化

導入・構築の容易性強化を目的として変更した項目を次の表に示します。

表 A–38　導入・構築の容易性強化を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
Web サービスプロバイダ側での web.xml の指定必須タグの変更	Web サービスプロバイダ側での web.xml で，listener タグ，servlet タグおよび servlet-mapping タグの指定を必須から任意に変更しました。	リファレンス 定義編 (サーバ定義)	2.2.3
論理サーバのネットワークリソース使用	J2EE アプリケーションからほかのホスト上にあるネットワークリソースやネットワークドライブにアクセスするための機能を追加しました。	機能解説 運用／監視／連携編	1.2.3，5.2，5.7

項目	変更の概要	参照先マニュアル	参照個所
サンプルプログラムの実行手順の簡略化	一部のサンプルプログラムを EAR 形式で提供することによって，サンプルプログラムの実行手順を簡略化しました。	ファーストステップガイド	3.5
		システム構築・運用ガイド	付録 L
運用管理ポータルの画面の動作の改善	画面の更新間隔のデフォルトを「更新しない」から「3 秒」に変更しました。	運用管理ポータル操作ガイド	7.4.1
セットアップウィザードの完了画面の改善	セットアップウィザード完了時の画面に，セットアップで使用した簡易構築定義ファイルおよび Connector 属性ファイルが表示されるようになりました。	システム構築・運用ガイド	2.2.6
仮想化環境の構築	ハイパーバイザを使用して実現する仮想サーバ上に，アプリケーションサーバを構築する手順を追加しました。	仮想化システム構築・運用ガイド	3 章，5 章

(2)　標準機能・既存機能への対応

標準機能・既存機能への対応を目的として変更した項目を次の表に示します。

表 A–39　標準機能・既存機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
OpenTP1 からの呼び出しへの対応	OpenTP1 からアプリケーションサーバ上で動作する Message-driven Bean を呼び出せるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	4 章
JMS への対応	JMS 1.1 仕様に対応した CJMS プロバイダ機能を使用できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	7 章
Java SE 6 への対応	Java SE 6 の機能が使用できるようになりました。	機能解説 保守／移行編	5.5, 5.8.1
ジェネリクスの使用への対応	EJB でジェネリクスを使用できるようになりました。	機能解説 基本・開発編 (EJB コンテナ)	4.2.18

(3)　信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A–40　信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
明示管理ヒープ機能の使用性向上	自動配置設定ファイルを使用して，明示管理ヒープ機能を容易に使用できるようになりました。	システム設計ガイド	7.2, 7.7.3, 7.11.4, 7.12.1
		機能解説 拡張編	7 章

項目	変更の概要	参照先マニュアル	参照個所
データベースセッションフェイルオーバ機能の URI 単位での抑止	データベースセッションフェイルオーバ機能を使用する場合に，機能の対象外にするリクエストを URI 単位で指定できるようになりました。	機能解説 拡張編	5.6.1
仮想化環境での障害監視	仮想化システムで，仮想サーバを監視し，障害の発生を検知できるようになりました。	仮想化システム構築・運用ガイド	－

(4) 運用性の維持・向上

運用性の維持・向上を目的として変更した項目を次の表に示します。

表 A-41 運用性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
管理ユーザアカウントの省略	運用管理ポータル，Management Server のコマンド，または Smart Composer 機能のコマンドで，ユーザのログイン ID およびパスワードの入力を省略できるようになりました。	システム構築・運用ガイド	4.1.15
		運用管理ポータル操作ガイド	2.2，7.1.1，7.1.2，7.1.3，8.1，8.2.1，付録 E.2
		リファレンス コマンド編	1.4，mngsvrctl（Management Server の起動／停止／セットアップ），mngsvrutil（Management Server の運用管理コマンド），8.3，cmx_admin_passwd（Management Server の管理ユーザアカウントの設定）
仮想化環境の運用	仮想化システムで，複数の仮想サーバを対象にした一括起動・一括停止，スケールイン・スケールアウトなどを実行する手順を追加しました。	仮想化システム構築・運用ガイド	4 章，6 章

(5) そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A-42 そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
Tenured 領域内不要オブジェクト統計機能	Tenured 領域内で不要となったオブジェクトだけを特定できるようになりました。	機能解説 保守／移行編	9.8
Tenured 増加要因の基点オブジェクトリスト出力機能	Tenured 領域内不要オブジェクト統計機能を使って特定した，不要オブジェクトの基点となるオブジェクトの情報を出力できるようになりました。		9.9
クラス別統計情報解析機能	クラス別統計情報を CSV 形式で出力できるようになりました。		9.10

項目	変更の概要	参照先マニュアル	参照個所
論理サーバの自動再起動回数オーバー検知によるクラスタ系切り替え	Management Server を系切り替えの監視対象としているクラスタ構成の場合，論理サーバが異常停止状態(自動再起動回数をオーバーした状態，または自動再起動回数の設定が 0 のときに障害を検知した状態)になったタイミングでの系切り替えができるようになりました。	機能解説 運用／監視／連携編	18.4.3, 18.5.3, 16.2.2, 16.3.3, 16.3.4
ホスト単位管理モデルを対象とした系切り替えシステム	クラスタソフトウェアと連携したシステム運用で，ホスト単位管理モデルを対象にした系切り替えができるようになりました。		16 章
ACOS（AX2000，BS320）のサポート	使用できる負荷分散機の種類に ACOS（AX2000，BS320）が追加になりました。	システム構築・運用ガイド	4.7.2, 4.7.3, 4.7.5, 4.7.6，付録 J，付録 J.2
		リファレンス 定義編(サーバ定義)	4.2.4, 4.3.2, 4.3.4, 4.3.5, 4.3.6, 4.7.1
CMT でトランザクション管理をする場合に Stateful Session Bean（SessionSynchronization）に指定できるトランザクション属性の追加	CMT でトランザクション管理をする場合に，Stateful Session Bean（SessionSynchronization）にトランザクション属性として Supports，NotSupported および Never を指定できるようになりました。	機能解説 基本・開発編(EJB コンテナ)	2.7.3
OutOfMemoryError 発生時の運用管理エージェントの強制終了	JavaVM で OutOfMemoryError が発生したときに，運用管理エージェントが強制終了するようになりました。	機能解説 保守／移行編	2.5.5
スレッドの非同期並行処理	TimerManager および WorkManager を使用して，非同期タイマ処理および非同期スレッド処理を実現できるようになりました。	機能解説 拡張編	－

付録 A.12　08-00 での主な機能変更

(1)　開発生産性の向上

開発生産性の向上を目的として変更した項目を次の表に示します。

表 A-43　開発生産性の向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
ほかのアプリケーションサーバ製品からの移行容易化	ほかのアプリケーションサーバ製品からの移行を円滑に実施するため，次の機能を使用できるようになりました。 • HTTP セッションの上限が例外で判定できるようになりました。 • JavaBeans の ID が重複している場合や，カスタムタグの属性名と TLD の定義で大文字・小文字が異なる場合に，トランスレーションエラーが発生することを抑止できるようになりました。	機能解説 基本・開発編 (Web コンテナ)	2.3, 2.7.5
cosminexus.xml の提供	アプリケーションサーバ独自の属性を cosminexus.xml に記載することによって，J2EE アプリケーションを J2EE サーバにインポート後，プロパティの設定をしないで開始できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	16.3

(2)　標準機能への対応

標準機能への対応を目的として変更した項目を次の表に示します。

表 A-44　標準機能への対応を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
Servlet 2.5 への対応	Servlet 2.5 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	2.2, 2.5.4, 2.6，8 章
JSP 2.1 への対応	JSP 2.1 に対応しました。	機能解説 基本・開発編 (Web コンテナ)	2.3.1, 2.3.3, 2.5, 2.6，8 章
JSP デバッグ	MyEclipse を使用した開発環境で JSP デバッグができるようになりました。※	機能解説 基本・開発編 (Web コンテナ)	2.4
タグライブラリのライブラリ JAR への格納と TLD のマッピング	タグライブラリをライブラリ JAR に格納した場合に，Web アプリケーション開始時に Web コンテナによってライブラリ JAR 内の TLD ファイルを検索し，自動的にマッピングできるようになりました。	機能解説 基本・開発編 (Web コンテナ)	2.3.4
application.xml の省略	J2EE アプリケーションで application.xml が省略できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	16.4
アノテーションと DD の併用	アノテーションと DD を併用できるようになり，アノテーションで指定した内容を DD で更新できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	17.5
アノテーションの Java EE 5 標準準拠（デフォルトインターセプタ）	デフォルトインターセプタをライブラリ JAR に格納できるようになりました。また，デフォルトインターセプタから DI できるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	16.4

項目	変更の概要	参照先マニュアル	参照個所
@Resource の参照解決	@Resource でリソースの参照解決ができるようになりました。	機能解説 基本・開発編 (コンテナ共通機能)	17.4
JPA への対応	JPA 仕様に対応しました。	機能解説 基本・開発編 (コンテナ共通機能)	6 章

注※　09-00 以降では，WTP を使用した開発環境で JSP デバッグ機能を使用できます。

(3)　信頼性の維持・向上

信頼性の維持・向上を目的として変更した項目を次の表に示します。

表 A–45　信頼性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
セッション情報の永続化	HTTP セッションのセッション情報をデータベースに保存して引き継げるようになりました。	機能解説 拡張編	5 章，6 章
FullGC の抑止	FullGC の要因となるオブジェクトを Java ヒープ外に配置することで，FullGC 発生を抑止できるようになりました。	機能解説 拡張編	7 章
クライアント性能モニタ	クライアント処理に掛かった時間を調査・分析できるようになりました。	－	－

（凡例）－：09-00 で削除された機能です。

(4)　運用性の維持・向上

運用性の維持・向上を目的として変更した項目を次の表に示します。

表 A–46　運用性の維持・向上を目的とした変更

項目	変更の概要	参照先マニュアル	参照個所
運用管理ポータルでのアプリケーション操作性向上	アプリケーションおよびリソースの操作について，サーバ管理コマンドと運用管理ポータルの相互運用ができるようになりました。	運用管理ポータル操作ガイド	1.1.3

(5)　そのほかの目的

そのほかの目的で変更した項目を次の表に示します。

表 A–47　そのほかの目的による変更

項目	変更の概要	参照先マニュアル	参照個所
無効な HTTP Cookie の削除	無効な HTTP Cookie を削除できるようになりました。	機能解説 基本・開発編 (Web コンテナ)	2.7.4

項目	変更の概要	参照先マニュアル	参照個所
ネーミングサービスの障害検知	ネーミングサービスの障害が発生した場合に，EJB クライアントが，より早くエラーを検知できるようになりました。	機能解説 基本・開発編(コンテナ共通機能)	2.9
コネクション障害検知タイムアウト	コネクション障害検知タイムアウトのタイムアウト時間を指定できるようになりました。	機能解説 基本・開発編(コンテナ共通機能)	3.15.1
Oracle11g への対応	データベースとして Oracle11g が使用できるようになりました。	機能解説 基本・開発編(コンテナ共通機能)	3 章
バッチ処理のスケジューリング	バッチアプリケーションの実行を CTM によってスケジューリングできるようになりました。	機能解説 拡張編	4 章
バッチ処理のログ	バッチ実行コマンドのログファイルのサイズ，面数，ログの排他処理失敗時のリトライ回数とリトライ間隔を指定できるようになりました。	リファレンス 定義編(サーバ定義)	3.2.5
snapshot ログ	snapshot ログの収集内容が変更されました。	機能解説 保守／移行編	付録 A.1，付録 A.2
メソッドキャンセルの保護区公開	メソッドキャンセルの対象外となる保護区リストの内容を公開しました。	機能解説 運用／監視／連携編	付録 C
統計前のガーベージコレクション選択機能	クラス別統計情報を出力する前に，ガーベージコレクションを実行するかどうかを選択できるようになりました。	機能解説 保守／移行編	9.7
Survivor 領域の年齢分布情報出力機能	Survivor 領域の Java オブジェクトの年齢分布情報を JavaVM ログファイルに出力できるようになりました。	機能解説 保守／移行編	9.11
ファイナライズ滞留解消機能	JavaVM のファイナライズ処理の状態を監視して，処理の滞留を解消できるようになりました。	－	－
サーバ管理コマンドの最大ヒープサイズの変更	サーバ管理コマンドが使用する最大ヒープサイズが変更されました。	リファレンス 定義編(サーバ定義)	5.2.1，5.2.2
推奨しない表示名を指定された場合の対応	J2EE アプリケーションで推奨しない表示名を指定された場合にメッセージが出力されるようになりました。	メッセージ(構築／運用／開発用)	KDJE42374-W

（凡例）－：09-00 で削除された機能です。

付録 B　例外リストの登録（Windows の場合）

Windows の場合に，ファイアウォールを有効にするときは，ファイアウォールの例外リストに構成ソフトウェアのプログラムを登録する必要があります。登録する例外リストは，構成ソフトウェアによって異なります。

次の表に示す構成ソフトウェアをインストールしている場合に，ファイアウォールを有効にするときは，コマンドプロンプトでコマンドを実行して，例外リストを登録してください。構成ソフトウェアごとに実行する例外リスト登録コマンドを次の表に示します。なお，アプリケーションサーバおよび BPM/ESB 基盤の製品を利用して作成したプログラムも，例外リストに追加する必要があります。例外リスト登録コマンドなどを使用して，作成したプログラムを例外リストに登録してください。

表 B-1　構成ソフトウェアごとに実行する例外リスト登録コマンド

インストール済みの構成ソフトウェア	例外リスト登録の実施条件	実行する例外リスト登録コマンド
Component Container	必ず実施	netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CC¥server¥bin¥cjstartsv.exe" name="Cosminexus Component Container" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CC¥web¥bin¥cjstartweb.exe" name="Cosminexus Component Container" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CC¥client¥bin¥cjclstartap.exe" name="Cosminexus Component Container" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥manager¥bin¥adminagent.exe" name="Cosminexus Component Container" mode=ENABLE
	サーバ管理コマンドを実行する場合	netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥vbj.exe" name="Cosminexus Component Container" mode=ENABLE※1
	バッチコマンドでスケジューリング機能を使用する場合	netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CC¥batch¥bin¥cjexecjob.exe" name="Cosminexus Component Container" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CC¥batch¥bin¥cjkilljob.exe" name="Cosminexus Component Container" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CC¥batch¥bin¥cjlistjob.exe" name="Cosminexus Component Container" mode=ENABLE

インストール済みの構成ソフトウェア	例外リスト登録の実施条件	実行する例外リスト登録コマンド
	仮想サーバのサーバ通信エージェントを使用する場合	netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥sinagent¥bin¥sinaviagent.exe" name="uCosminexus SI Navigation System Agent" mode=ENABLE
Component Transaction Monitor	必ず実施	netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmchpara.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmd.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmdmd.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmdmstart.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmdmstop.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmgetior.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmholdque.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmidl2cpp.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmidl2j.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmlsque.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmnaminfo.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmregltd.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE

インストール済みの構成ソフトウェア	例外リスト登録の実施条件	実行する例外リスト登録コマンド
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmridinfo.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmrlesque.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmstart.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmstop.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmstartgw.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmstopgw.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥CTM¥bin¥ctmtscgwd.exe" name="Cosminexus Component Transaction Monitor" mode=ENABLE
HTTP Server	必ず実施	netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥httpsd¥httpsd.exe" name="Cosminexus HTTP Server"mode=ENABLE
TPBroker※2	必ず実施	netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥events.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥gatekeeper.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥irep.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥nameserv.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥oad.exe" name="Cosminexus TPBroker" mode=ENABLE

インストール済みの構成ソフトウェア	例外リスト登録の実施条件	実行する例外リスト登録コマンド
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥osagent.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥osfind.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥admd.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥otsd.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥trnctxsv.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥tsstoptrnctxsv.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥tscommit.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥tslstrn.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥tsrollback.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥tsstat.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥tsstop.exe" name="Cosminexus TPBroker" mode=ENABLE
		netsh firewall add allowedprogram program="<Application Server のインストールディレクトリ>¥TPB¥bin¥tstrnsts.exe" name="Cosminexus TPBroker" mode=ENABLE
HiRDB/Single Server Version 10	Developer または Service Architect で組み込みデータ	for %%p in (<Developer または Service Architect のインストールディレクトリ>¥DB¥bin¥*.exe) do netsh firewall set allowedprogram %%p "Cosminexus Developer(DB)"

インストール済みの構成ソフトウェア	例外リスト登録の実施条件	実行する例外リスト登録コマンド
	ベースを使用する場合	for %%p in (<Developer または Service Architect のインストールディレクトリ>¥DB¥lib¥servers¥*.exe) do netsh firewall set allowedprogram %%p "Cosminexus Developer(DB)"
		for %%p in (<Developer または Service Architect のインストールディレクトリ>¥DB¥SAMPLE¥sampleconf¥*.exe) do netsh firewall set allowedprogram %%p "Cosminexus Developer(DB)"
		for %%p in (<Developer または Service Architect のインストールディレクトリ>¥DB¥SAMPLE¥tools¥*.exe) do netsh firewall set allowedprogram %%p "Cosminexus Developer(DB)"
		for %%p in (<Developer または Service Architect のインストールディレクトリ>¥DB¥PDISTUP¥bin¥*.exe) do netsh firewall set allowedprogram %%p "Cosminexus Developer(DB)"

注※1　vbj コマンドを使用している EJB クライアントも，ファイアウォールのフィルタリング対象から除外されるので，注意してください。

注※2　tssetfw コマンドを使用して例外リストを登録することもできます。tssetfw コマンドについては，「TPBroker 追加機能取扱説明書」を参照してください。

付録 C　用語解説

マニュアルで使用する用語について

マニュアル「アプリケーションサーバ & BPM/ESB 基盤 用語解説」を参照してください。

索引

記号

<LB 接続情報の識別名>.properties　475
<security-constraint>要素の設定　36
<security-identity>要素の設定　36
<ua:attributeEntries>Entries</ua:attributeEntries>タグ　455
<ua:attributeEntry/>タグ　456
<ua:chpw/>タグ　456
<ua:exception>Body</ua:exception>タグ　458
<ua:getAttribute/>タグ　459
<ua:getAttributeNames/>タグ　461
<ua:getAttributes/>タグ　460
<ua:getPrincipalName/>タグ　458
<ua:login/>タグ　462
<ua:logout/>タグ　464
<ua:notLogin>Body</ua:notLogin>タグ　464
「リソース監視」で出力される JDBC 接続の障害情報　291
「リソース監視」で出力される LDAP 接続の障害情報　283
「リソース監視」のサーバビューの構成　271
「リソース監視」のツリーペインの構成　271
「リソース監視」のホストビューの構成　271
「リポジトリ管理」での注意事項　240
「リポジトリ管理」のツリーペインの構成　243
-nosecurity オプション　37

数字

09-70 での主な機能変更　537
09-80 での主な機能変更　536
09-87 での主な機能変更　536
11-00 での主な機能変更　534
11-10 での主な機能変更　533

A

Active Directory を使用する場合の設定　170
addAttribute メソッド　392, 398
addSSODataListener メソッド　340
addSSOData メソッド　339
addUserData メソッド（形式 1）　351
addUserData メソッド（形式 2）　352
API と SPI の関係　103
API による実装時の注意事項　148
API の例外クラス　451
API 用パラメタ　318
API を使用したユーザ認証の実装　144
Application〔JAAS のコンフィグレーションファイルのオプション〕　306
AttributeEntry クラス　331
AttributeEntry コンストラクタ　332

C

ChangeDataFailedException クラス　336
ChangeDataFailedException コンストラクタ　336
changePassword メソッド　371
check メソッド（形式 1）　363
check メソッド（形式 2）　364
close メソッド　359
com.cosminexus.admin.auth.api.repository.event.ChangeDataFailedException　452
com.cosminexus.admin.auth.api.repository.event.SSODataListenerException　452
com.cosminexus.admin.auth.api.repository.ldap.config〔API 用パラメタ〕　318
com.cosminexus.admin.auth.api.repository.ldap.ObjectClassError　452
com.cosminexus.admin.auth.CryptoException　452
com.cosminexus.admin.auth.custom.lm〔DelegationLoginModule に指定するオプション〕　309
com.cosminexus.admin.auth.custom.modules〔カスタムログインモジュールのパラメタ〕　320
com.cosminexus.admin.auth.gsession.keep_password〔WebPasswordJDBCLoginModule に指定するオプション〕　311

com.cosminexus.admin.auth.gsession.keep_password〔WebPasswordLDAPLoginModule に指定するオプション〕 312
com.cosminexus.admin.auth.gsession.keep_password〔WebPasswordLoginModule に指定するオプション〕 308
com.cosminexus.admin.auth.gsession.keep_password〔標準ログインモジュールのパラメタ〕 320
com.cosminexus.admin.auth.jdbc.conn.password〔JDBC 用パラメタ〕 316
com.cosminexus.admin.auth.jdbc.conn.retry.count〔JDBC 用パラメタ〕 317
com.cosminexus.admin.auth.jdbc.conn.retry.wait〔JDBC 用パラメタ〕 317
com.cosminexus.admin.auth.jdbc.conn.url〔JDBC 用パラメタ〕 316
com.cosminexus.admin.auth.jdbc.conn.user〔JDBC 用パラメタ〕 316
com.cosminexus.admin.auth.jdbc.driver 316
com.cosminexus.admin.auth.jdbc.password.encrypt.ex〔JDBC 用パラメタ〕 318
com.cosminexus.admin.auth.jdbc.password.encrypt〔JDBC 用パラメタ〕 317
com.cosminexus.admin.auth.jdbc.password.type〔JDBC 用パラメタ〕 317
com.cosminexus.admin.auth.jdbc.pool.enable〔JDBC 用パラメタ〕 316
com.cosminexus.admin.auth.jdbc.pool.gc_interval〔JDBC 用パラメタ〕 316
com.cosminexus.admin.auth.jdbc.pool.max_spare〔JDBC 用パラメタ〕 316
com.cosminexus.admin.auth.jdbc.pool.max〔JDBC 用パラメタ〕 316
com.cosminexus.admin.auth.jdbc.pool.min_spare〔JDBC 用パラメタ〕 316
com.cosminexus.admin.auth.jdbc.r〔WebPasswordJDBCLoginModule に指定するオプション〕 310
com.cosminexus.admin.auth.jdbc.sql〔JDBC 用パラメタ〕 317
com.cosminexus.admin.auth.keep_password.encrypt〔WebPasswordJDBCLoginModule に指定するオプション〕 310
com.cosminexus.admin.auth.keep_password.encrypt〔WebPasswordLDAPLoginModule に指定するオプション〕 312
com.cosminexus.admin.auth.keep_password.encrypt〔WebPasswordLoginModule に指定するオプション〕 308
com.cosminexus.admin.auth.keep_password.encrypt〔標準ログインモジュールのパラメタ〕 320
com.cosminexus.admin.auth.keep_password〔WebPasswordJDBCLoginModule に指定するオプション〕 310
com.cosminexus.admin.auth.keep_password〔WebPasswordLDAPLoginModule に指定するオプション〕 312
com.cosminexus.admin.auth.keep_password〔WebPasswordLoginModule に指定するオプション〕 308
com.cosminexus.admin.auth.keep_password〔標準ログインモジュールのパラメタ〕 320
com.cosminexus.admin.auth.ldap.attr.password〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.attr.userid〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.basedn〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.certificate.attr.userid〔JNDI 用パラメタ〕 315
com.cosminexus.admin.auth.ldap.conn.read_timeout〔JNDI 用パラメタ〕 315
com.cosminexus.admin.auth.ldap.conn.retry.count〔JNDI 用パラメタ〕 315
com.cosminexus.admin.auth.ldap.conn.retry.wait〔JNDI 用パラメタ〕 315
com.cosminexus.admin.auth.ldap.directory.kind〔JNDI 用パラメタ〕 315
com.cosminexus.admin.auth.ldap.password.encrypt.ex〔JNDI 用パラメタ〕 315
com.cosminexus.admin.auth.ldap.password.encrypt〔JNDI 用パラメタ〕 315
com.cosminexus.admin.auth.ldap.pool.enable〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.pool.gc_interval〔JNDI 用パラメタ〕 314

com.cosminexus.admin.auth.ldap.pool.max_spare〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.pool.max〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.pool.min_spare〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.r〔WebCertificateLoginModule に指定するオプション〕 310
com.cosminexus.admin.auth.ldap.r〔WebPasswordLDAPLoginModule に指定するオプション〕 311
com.cosminexus.admin.auth.ldap.r〔WebPasswordLoginModule に指定するオプション〕 307
com.cosminexus.admin.auth.ldap.search.scope〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.search.userrdn〔JNDI 用パラメタ〕 314
com.cosminexus.admin.auth.ldap.w〔WebPasswordLDAPLoginModule に指定するオプション〕 311
com.cosminexus.admin.auth.ldap.w〔WebPasswordLoginModule に指定するオプション〕 307
com.cosminexus.admin.auth.param_check.enable〔標準ログインモジュールのパラメタ〕 320
com.cosminexus.admin.auth.realm〔DelegationLoginModule に指定するオプション〕 309
com.cosminexus.admin.auth.realm〔WebCertificateLoginModule に指定するオプション〕 310
com.cosminexus.admin.auth.realm〔WebPasswordJDBCLoginModule に指定するオプション〕 310
com.cosminexus.admin.auth.realm〔WebPasswordLDAPLoginModule に指定するオプション〕 312
com.cosminexus.admin.auth.realm〔WebPasswordLoginModule に指定するオプション〕 308
com.cosminexus.admin.auth.realm〔WebSSOLoginModule に指定するオプション〕 309
com.cosminexus.admin.auth.sfo.disable〔セッションフェイルオーバ対応のパラメタ〕 321
com.cosminexus.admin.auth.sso.encrypt〔シングルサインオン用パラメタ〕 319
com.cosminexus.admin.auth.sso.keyfile〔シングルサインオン用パラメタ〕 319
com.cosminexus.admin.auth.sso.ldap.r〔WebSSOLoginModule に指定するオプション〕 309
com.cosminexus.admin.auth.sso.ldap.r〔シングルサインオン用パラメタ〕 319
com.cosminexus.admin.auth.sso.ldap.w〔WebPasswordLDAPLoginModule に指定するオプション〕 311
com.cosminexus.admin.auth.sso.ldap.w〔WebPasswordLoginModule に指定するオプション〕 307
com.cosminexus.admin.auth.sso.ldap.w〔WebSSOLoginModule に指定するオプション〕 309
com.cosminexus.admin.auth.sso.ldap.w〔シングルサインオン用パラメタ〕 319
com.cosminexus.admin.auth.sso.lm〔シングルサインオン用パラメタ〕 319
com.cosminexus.admin.auth.sso.param.pubdat〔シングルサインオン用パラメタ〕 319
com.cosminexus.admin.auth.sso.param.secdat〔シングルサインオン用パラメタ〕 319
com.cosminexus.admin.auth.sso.param.userid〔シングルサインオン用パラメタ〕 319
com.cosminexus.admin.auth.sso〔WebSSOLoginModule に指定するオプション〕 309
com.cosminexus.admin.auth.trace.level〔トレースのパラメタ〕 321
com.cosminexus.admin.auth.trace.prefix〔トレースのパラメタ〕 321
com.cosminexus.admin.auth.trace.rotate〔トレースのパラメタ〕 321

com.cosminexus.admin.auth.trace.size〔トレースのパラメタ〕 321
com.cosminexus.admin.common.ConfigError 452
com.cosminexus.admin.common.FormatError 452
com.cosminexus.admin.common.ParameterError 452
com.cosminexus.admin.common.UAException 451
com.sun.jndi.ldap.connect.timeout〔JNDI 用パラメタ〕 315
convpw 295
cosminexus.xml を含まないアプリケーションのプロパティ設定 31
CSV 形式ファイルの記述例 325
CSV 形式ファイルの基本仕様 322

D

DD の設定による Web コンテナのユーザ認証 192
DD の設定による Web コンテナのユーザ認証の機能 192
DelegationLoginModule 121
DelegationLoginModule クラス 337
DelegationLoginModule に指定するオプション 309

E

EJB クライアントアプリケーションでのセキュリティの実装 197
EJB クライアントアプリケーションの実装で使用する API 466
encrypt メソッド 370
Enterprise Bean のセキュリティアイデンティティ 227
Enterprise Bean のセキュリティロールリファレンスの定義 221

F

Flag〔JAAS のコンフィグレーションファイルのオプション〕 307

G

getAlias メソッド 332
getAttributeEntries メソッド 404, 423
getAttributeNames メソッド 393, 399
getAttributeName メソッド 333
getAttributes メソッド 394, 400
getAttribute メソッド 392, 399
getException メソッド 389
getListeners メソッド 389
getLoginInfoManager メソッド 468
getMappingRealms メソッド 375
getMapping メソッド 375
getName メソッド 423
getObjectClasses メソッド 367
getOldPublicData メソッド 381
getOldSecretData メソッド 382
getOption メソッド 424
getPassword メソッド 424
getPublicData メソッド 376, 382
getRequest メソッド 405, 425, 440
getResponse メソッド 405, 425, 441
getSecretData メソッド 383
getSession メソッド 416
getSSODataListeners メソッド 342
getSSOData メソッド 341
getSubcontext メソッド 333, 368
getSubjectID メソッド 406
getTagEntry メソッド 406, 426, 441
getTagID メソッド 407, 426, 442
getUserData メソッド 354
getUserId メソッド 383
getUserID メソッド 416

H

handle メソッド 413, 420, 434, 446
hasMoreElements メソッド 361
hasMore メソッド 360
HTTP Server の SSL の設定 203

I

IDS　82

J

JAAS　102
jaas.conf　306
jaas.conf の再読み込み　178
jaas.conf の作成　177
jaas.conf の設定例　181
JAAS 対応ユーザ管理　103
JAAS でのユーザ認証の有効期限　109
JAAS のコンフィグレーションファイル　306
JAAS のコンフィグレーションファイルの定義例 132
JAAS のログインモジュールの例外クラス　449
java.naming.provider.url〔JNDI 用パラメタ〕 313
java.naming.security.credentials〔JNDI 用パラメタ〕　313
java.naming.security.principal〔JNDI 用パラメタ〕 313
JavaVM のプロパティの設定　187
javax.net.ssl.trustStorePassword〔負荷分散機定義プロパティファイルのキー〕　477, 480
javax.net.ssl.trustStore〔負荷分散機定義プロパティファイルのキー〕　477, 479
javax.security.auth.login.AccountExpiredException　449
javax.security.auth.login.CredentialExpiredException　449
javax.security.auth.login.FailedLoginException 449
javax.security.auth.login.LoginException　449, 450
Java 標準仕様（JAAS）に準拠したユーザ認証　102
JDBC 接続の障害情報の表示　291
JDBC 接続の定義情報の表示　288
JDBC 接続プールの空き待ち監視のリセット　287
JDBC 接続プールモニタの表示　285
JDBC 接続モニタの監視　285
JNDI 用パラメタ　313
JSP タグライブラリ　141

L

lb.ACOS.privilegedexec.password〔負荷分散機接続設定プロパティファイルのキー〕　477, 479
lb.API.protocol〔負荷分散機接続設定プロパティファイルのキー〕　477, 479
lb.host〔負荷分散機定義プロパティファイルのキー〕 476, 478
lb.password〔負荷分散機定義プロパティファイルのキー〕　476, 479
lb.persistence.cookie-insert.templatename〔負荷分散機定義プロパティファイルのキー〕　476, 479
lb.port〔負荷分散機定義プロパティファイルのキー〕 476, 479
lb.properties　473
lb.protocol〔負荷分散機定義プロパティファイルのキー〕　476, 479
lb.timeout〔負荷分散機定義プロパティファイルのキー〕　476, 479
lb.type〔負荷分散機定義プロパティファイルのキー〕 476, 478
lb.user〔負荷分散機定義プロパティファイルのキー〕 476, 479
LdapSSODataManager クラス　338
LdapSSODataManager コンストラクタ　339
LdapUserDataManager クラス　348
LdapUserDataManager コンストラクタ　349
LdapUserEnumeration インタフェース　359
LDAP 接続の障害情報の表示　283
LDAP 接続の定義情報の表示　279
LDAP 接続プールの空き待ち監視のリセット　278
LDAP 接続プールモニタの表示　276
LDAP 接続モニタの監視　276
LDAP ディレクトリサーバの設置　165
LDAP ディレクトリサーバの設定　165
LDAP ディレクトリサーバの多重化による接続フェイルオーバ　138
LDAP ディレクトリサーバの多重化の構成例　139
LDAP ディレクトリサーバの多重化の構成例（マルチマスタ構成）　140
LDAP ディレクトリサーバへのユーザ情報の登録 138
LINK_xxxx　324

listUsers メソッド（形式 1） 342, 355
listUsers メソッド（形式 2） 343, 355
LoginInfoManager クラス 467
LoginModule インタフェースの実装時の留意点 155
LoginUtil クラス 363
login メソッド 468
logout メソッド 469

M

modifySSOData メソッド 344
modifyUserData メソッド 356
ModuleOptions〔JAAS のコンフィグレーションファイルのオプション〕 307

N

nextElement メソッド 362
next メソッド 361

O

ObjectClassEntry クラス 366
ObjectClassEntry コンストラクタ 366
OPERATION 324

P

PasswordCryptography インタフェース 370
PasswordUtil クラス 371
Principal インタフェース 373
Principal オブジェクトの実装時の留意点 156
PUBLICDATA 324

R

Realm 101
REALMNAME 324
removeAttribute メソッド 395, 400
removeMapping メソッド 376
removeSSODataListener メソッド 346
removeSSOData メソッド 345
removeUserData メソッド 357
Run as 機能 196

S

SECRETDATA 324
SecurityManager の機能による J2EE サーバの実行時の保護 36
setAlias メソッド 334
setAttributeEntries メソッド 407, 427
setAttributeName メソッド 334
setException メソッド 390
setMapping メソッド 377
setName メソッド 427
setObjectClasses メソッド 368
setOption メソッド 428
setPassword メソッド 401, 429
setPublicData メソッド 378
setRequest メソッド 408, 429, 442
setResponse メソッド 408, 430, 443
setSecretData メソッド 378
setSession メソッド 417
setSubcontext メソッド 335, 369
setSubjectID メソッド 409
setTagEntry メソッド 410, 430, 443
setTagID メソッド 410, 431, 444
setUserID メソッド 417
size メソッド 395, 401
SSL アクセラレータ 35, 94
SSL 使用による認証情報とデータの暗号化 202
SSL による暗号化 36
ssoDataAdded メソッド 385
SSODataEvent クラス 380
SSODataEvent コンストラクタ 380
SSODataListenerException クラス 388
SSODataListenerException コンストラクタ 388
SSODataListener インタフェース 384
ssoDataModified メソッド 386
ssoDataRemoved メソッド 386
SSOData クラス 374
SSOData コンストラクタ 374
ssoexport 296
ssogenkey 298

ssoimport　299

T

tierlb.properties　478

U

ua.conf　313
ua.conf の作成　179
ua.conf の設定例　182
ua.conf の設定例（UNIX の場合）　183
uachpw　302
uachpw コマンド　180
UserAttributes インタフェース　391
UserData クラス　397
UserData コンストラクタ　397
USERID　324

W

WebCertificateCallback クラス　403
WebCertificateCallback コンストラクタ　404
WebCertificateHandler クラス　411
WebCertificateHandler コンストラクタ　411
WebCertificateLoginModule　116
WebCertificateLoginModule クラス　414
WebCertificateLoginModule に指定するオプション　310
WebLogoutCallback クラス　415
WebLogoutCallback コンストラクタ　415
WebLogoutHandler クラス　419
WebLogoutHandler コンストラクタ　419
WebPasswordCallback クラス　421
WebPasswordCallback コンストラクタ　422
WebPasswordHandler クラス　432
WebPasswordHandler コンストラクタ　432
WebPasswordJDBCLoginModule　119
WebPasswordJDBCLoginModule クラス　436
WebPasswordJDBCLoginModule に指定するオプション　310
WebPasswordLDAPLoginModule　117
WebPasswordLDAPLoginModule クラス　437
WebPasswordLDAPLoginModule に指定するオプション　311
WebPasswordLoginModule　115
WebPasswordLoginModule クラス　438
WebSSOCallback クラス　439
WebSSOCallback コンストラクタ　440
WebSSOHandler クラス　445
WebSSOHandler コンストラクタ　445
WebSSOLoginModule　122
WebSSOLoginModule クラス　448
WebSSOLoginModule に指定するオプション　309
Web サーバの認証機能　202
Web サービスセキュリティの機能による SOAP メッセージの暗号化　36

あ

アプリケーションサーバ 11-20 での主な機能変更　32
アプリケーションサーバの機能　19
アプリケーション集中型の構成　86
アプリケーションで利用できる認証機能　95
アプリケーションの実行基盤としての機能　22
アプリケーションの実行基盤を運用・保守するための機能　23
アプリケーションの設定による認証　190
アプリケーション分散型の構成　90
暗号鍵ファイルの作成　174, 298
暗号鍵ファイルの作成（シングルサインオンを使用する場合）　174
暗号鍵ファイルの設定　248
暗号鍵ファイルの変更　174
暗号拡張　147

う

運用管理ポータルによる統合ユーザ管理の運用　231
運用管理ポータルによるリポジトリ管理（統合ユーザ管理）　241

お

オブジェクトクラスとユーザ定義属性の拡張　166

か

カスタムログインモジュール　136, 143
カスタムログインモジュールの実装時の留意点　155
カスタムログインモジュールの実装例　156
カスタムログインモジュールのセッション　130
カスタムログインモジュールのパラメタ　320
カスタムログインモジュールの呼び出し　137
カスタムログインモジュールを使用したユーザ認証の実装　154
仮想サーバマネージャ側の負荷分散機接続設定プロパティファイル　475

き

機能とマニュアルの対応　24
機能の分類　20

く

クライアント認証〔Web サーバの機能〕　202

こ

このマニュアルに記載している機能の説明　30
コマンドによる登録　168, 175
コンテナセキュリティとアクセス権管理　192
コンフィグレーションファイルの作成　177
コンフィグレーションファイルの設定例　180

さ

サーバ管理コマンドによるセキュリティロールとアプリケーションの操作　215
サーバ認証〔Web サーバの機能〕　202
サーブレットと JSP のセキュリティアイデンティティ　229
サーブレットと JSP のセキュリティロールリファレンスの定義　222
作業手順書　67

し

システムの目的と機能の対応　27
シングルサインオン　134
シングルサインオン情報リポジトリ　103
シングルサインオン情報リポジトリに対するシングルサインオン用のユーザ情報の編集　238
シングルサインオン情報リポジトリの DIT 構造　108
シングルサインオン情報リポジトリの参照　296
シングルサインオン情報リポジトリの登録　299
シングルサインオンへの対応例（カスタムログインモジュールの場合）　186
シングルサインオンへの対応例（標準ログインモジュールの場合）　185
シングルサインオン用認証情報の CSV 形式ファイル　322
シングルサインオン用パラメタ　318
シングルサインオンライブラリ　141
侵入検知システム　35, 82

せ

セキュリティアイデンティティ　196
セキュリティアイデンティティの機能　196
セキュリティアイデンティティの設定　227
セキュリティアイデンティティを使用した認証　196
セキュリティアイデンティティを使用した認証の設定　199
セキュリティの定義（セキュリティアイデンティティ）　227
セキュリティの定義（メソッドパーミッション）　224
セキュリティロールの設定　217
セキュリティロールのリファレンス定義　221
セッションの種類　130

そ

操作する画面（統合ユーザ管理のリソース監視）　234
操作する画面（統合ユーザ管理のリポジトリ管理）　239
属性の一覧の設定例　181

た

タグライブラリのタグの一覧　454
タグライブラリを使用したユーザ認証の実装　150

て

ティア側の負荷分散機接続設定プロパティファイル 478

と

統合ユーザ管理機能の設定手順 161
統合ユーザ管理で使用するコマンド 293
統合ユーザ管理で使用するコマンドの一覧 294
統合ユーザ管理で使用するコマンドの詳細 295
統合ユーザ管理で使用するファイル 304
統合ユーザ管理で使用するファイルの一覧 305
統合ユーザ管理で使用するユーザ情報の管理方法 105
統合ユーザ管理による認証 98
統合ユーザ管理のコンフィグレーションファイル 313
統合ユーザ管理の処理の流れ 111
統合ユーザ管理のセッション 130
統合ユーザ管理のセッション管理 130
統合ユーザ管理のセッションに登録したユーザ ID の削除 131
統合ユーザ管理のセッションの停止 275
統合ユーザ管理のリソース監視で確認できる項目 233
統合ユーザ管理のリソースの監視 233
統合ユーザ管理フレームワーク 36, 100, 143
統合ユーザ管理フレームワークが提供する API 141
統合ユーザ管理フレームワークで使用する API 328
統合ユーザ管理フレームワークで使用するタグライブラリ 453
統合ユーザ管理フレームワークと個別のユーザ管理の関連 102
統合ユーザ管理フレームワークによるユーザ認証の実装 143
統合ユーザ管理フレームワークのライブラリ 141
統合ユーザ管理フレームワークのライブラリによる登録 169, 175
統合ユーザ管理フレームワークのリポジトリの DIT 構造 106
統合ユーザ管理を使用する場合の処理の流れ 111
登録するユーザ情報の文法 169, 176

に

認証機能 36
認証機能を併用するときの注意 194
認証状態の引き継ぎ 109
認証処理 101
認証に成功した Subject を HttpSession に登録する実装 146
認証用プログラムのコーディング例（Windows の場合） 184

は

バインド情報の設定 244
パスワード認証時の暗号拡張 125
パスワード認証時の暗号拡張の実装 147
パスワードの暗号化 295
パスワードの変更 302

ひ

標準ログインモジュール 103, 104, 113
標準ログインモジュールのパラメタ 320

ふ

ファイアウォール 35, 82
ファイルのデプロイ 189
負荷分散機定義プロパティファイル 473
プログラムセキュリティ 193

め

メソッドパーミッションの設定方法 224

ゆ

ユーザ ID の取得の実装（API を使用する場合） 144
ユーザエントリの検索 259
ユーザエントリの構造 109
ユーザエントリの削除 267
ユーザエントリの作成 254
ユーザエントリの作成（シングルサインオン用） 257
ユーザエントリのスキーマ定義 249
ユーザエントリのスキーマ定義（シングルサインオン用） 252

ユーザエントリの編集　262
ユーザエントリの編集（シングルサインオン用）　264
ユーザ情報の管理　138
ユーザ情報の登録　168
ユーザ情報の登録（シングルサインオンを使用する場合）　175
ユーザ情報リポジトリ　103
ユーザ情報リポジトリに対するユーザ情報の編集 235
ユーザ情報リポジトリの DIT 構造　106
ユーザ情報を取得するための定義ファイル　322
ユーザ情報を追加および変更するための定義ファイル　323
ユーザ認証の有効期間　109
ユーザ認証ライブラリ　141
ユーザ認証ライブラリおよびシングルサインオンライブラリの位置づけ　142
ユーザ認証リポジトリ　101
ユーザの設定　217
ユーザの登録とアクセス権の設定　165
ユーザマッピング　101, 102
ユーザマッピング機能　104
ユーザマッピングと認証情報の定義ファイル　324

ら

ラインオペレーション　326

り

リソース監視（統合ユーザ管理）　269
リバースプロキシサーバ　35
リポジトリ管理　244
リポジトリ管理によるユーザ情報の編集　235

れ

例外クラス〔統合ユーザ管理フレームワークで使用する API〕　449
レルム　100, 101
レルムの削除　261
レルムの作成　246
レルム名　101
レルム名の決定　164

ろ

ロールにユーザを登録　218
ロールの設定　217
ロールの登録　217
ログアウトの実装（API を使用する場合）　147
ログインしたユーザ ID の登録　130
ログインしているかどうかを確認する方法　151
ログインセッションの監視　273
ログインセッションモニタの表示　273
ログインの実装（API を使用する場合）　144
ログインモジュールが使用するコンフィグレーションファイルのパラメタ　126
ログインモジュール名〔JAAS のコンフィグレーションファイルのオプション〕　306