セキュリティのための実行可能なオペレーション・リソースのリファレンス

Version 2024.1
2024-06-03

InterSystems Corporation　1 Memorial Drive　Cambridge MA 02142　www.intersystems.com

セキュリティのための実行可能なオペレーション･リソースのリファレンス
InterSystems IRIS Data Platform　Version 2024.1　2024-06-03

インターシステムズ社の製品に関するサポートやご質問は、以下にお問い合わせください：

InterSystems Worldwide Response Center (WRC)
Tel:　+1-617-621-0700
Tel:　+44 (0) 844 854 2917
Email:　support@InterSystems.com

目次

図一覧

テーブル一覧

インスタンスの保護

インスタンスの保護の概要

インスタンスの保護の概要

セキュリティを確保するには、インスタンス内と、インスタンスが属するより大きな環境でのアクションが必要です。そのため、InterSystems IRIS にはインスタンスの保護に役立つガイドとツールの両方が用意されています。以下はその概要です。

- InterSystems IRIS の導入を準備する際に確認するトピックのチェックリスト
- 組み込み機能を使用するインスタンスを保護するためのガイドとツール
- オペレーティング・システム・レベルで、インスタンスのプロセスを管理することによってインスタンスのセキュリティを強化するためのチェックリスト

インターシステムズのセキュリティのための準備

インターシステムズのセキュリティのための準備

ここでは、InterSystems IRIS をインストールする前に検討する必要があるセキュリティ関連の問題をいくつか取り上げて説明します。インターシステムズのセキュリティ機能の概要は、“インターシステムズのセキュリティについて” を参照してください。認証や承認についても詳細を確認できます。

このセクションでは、以下のトピックについて説明します。

- インターシステムズの初期セキュリティ設定 ー 各種既定のセキュリティ設定の特徴について説明します。特に、通常またはロック・ダウンのインターシステムズのセキュリティを使用する場合に役立ちます。
- ユーザ・アカウントの構成 ー InterSystems IRIS を実行するユーザ・アカウントに必要な許可について説明します。
- Kerberos を使用したセキュリティ環境の準備 ー InterSystems IRIS で認証メカニズムとして Kerberos を使用する予定の場合に実行する必要がある追加のタスクを詳細に説明します。お使いの環境で Kerberos を使用しない場合、このトピックは省略してかまいません。

重要　　このドキュメントで説明する環境よりさらに複雑なセキュリティ環境を使用する場合、具体的な設定方法については、インターシステムズのサポート窓口までお問い合わせください。

″インターシステムズのセキュリティについて″ の説明を参照し、このセクションの手順を実行したうえで、″インストール・ガイド″ で説明されているように、インストール手順に必要なセキュリティ情報を指定してください。

インターシステムズの初期セキュリティ設定

インストール時に、**[最小]**、**[通常]**、または **[ロック・ダウン]** の 3 つの中から初期セキュリティ構成を選択します。概して、プロダクション環境で使用するインスタンスには [ロック・ダウン]、開発環境で使用するインスタンスには [通常] を選択することをお勧めします。以下のセクションでは、これらの各構成の違い、および各構成の初期のサービス・プロパティについて説明します。

- 初期のユーザ・セキュリティ設定
- 初期のユーザ・アカウント・パスワード
- 初期のサービス・プロパティ

プロダクション環境の場合、選択するオプションにかかわらず、インストール後に個々のセキュリティ設定を調整する必要があります。詳細は、以下のセクションを参照してください。

- インスタンスのセキュリティの強化
- セキュリティ・アドバイザ
- インターシステムズのプロセスおよびオペレーティング・システム・リソースの保護
- 導入環境のセキュリティを強化するためのチェックリスト

重要　　メモリ・イメージ内のデータの可視性 (コア・ダンプと呼ばれることが多い) に懸念がある場合は、″メモリ・イメージに存在する機密データの保護″ を参照してください。

初期のユーザ・セキュリティ設定

InterSystems IRIS ユーザ・アカウントの一般情報は、″ユーザ・アカウント″ を参照してください。

すべてのユーザ・アカウントで特定のパスワード要件および設定が共有されます。次の表に示しているように、これらの設定の初期値は選択したセキュリティ・レベルに基づきます。

セキュリティ設定	最小	通常	ロック・ダウン	説明
パスワード・パターン*	3.32ANP	3.32ANP	8.32ANP	既定で、パスワードには英数字および句読点を使用できます。長さの初期要件は、最小および通常のインストールの場合は 3 ～ 32 文字、ロック・ダウン・インストールの場合は 8 ～ 32 文字です。 パスワード・パターンの詳細は、"パスワードの強固さとパスワードのポリシー" を参照してください。
不活動上限*	0	90 日間	90 日間	アカウントがアクティブではなかった期間が、この値で指定された日数を超えると、このアカウントは無効になります。最小のインストールでは、この上限は 0 に設定されます。これは、アカウントがアクティブではない期間がどれだけ続いても、アカウントは無効化されないことを表します。通常のインストールやロック・ダウン・インストールに対する上限の既定値は 90 日間です。
_SYSTEM ユーザの有効化	可	可	不可	
UnknownUser に割り当てられるロール	%All	なし	なし	認証されていないユーザが接続した場合、InterSystems IRIS では、UnknownUser という特殊な名前が $USERNAME に割り当てられ、そのユーザに対して定義されているロールが $ROLES に割り当てられます。最小セキュリティのインストールでは、UnknownUser は %All ロールに割り当てられます。最小以外のセキュリティ・レベルを選択した場合、UnknownUser にはロールは割り当てられません。 $USERNAME および $ROLES の使用法の詳細は、"ユーザ" および "ロール" を参照してください。

* これらの設定は、管理ポータルの **[システム]→[セキュリティ管理]→[システムセキュリティ設定]→[システムワイドセキュリティパラメータ]** ページから管理できます。詳細は、"システム規模のセキュリティ・パラメータ" を参照してください。

初期のユーザ・アカウント・パスワード

InterSystems IRIS では、インストール時に複数のユーザ・アカウントが作成されます。事前定義の InterSystems IRIS ユーザ・アカウントが既定で備えるパスワードおよび動作内容は、インストールが最小のセキュリティ、通常のセキュリティ、またはロック・ダウン・セキュリティのいずれの条件で実行されたかによって異なります。これらの相違点は、以下のとおりです。

- 最小セキュリティ – 作成されたすべてのアカウントのうち、_PUBLIC を除くアカウントで、最初の既定のパスワードが“SYS”に設定されます。InterSystems IRIS インスタンスに対する承認されないアクセスを防止するために、UnknownUser を除くすべてのアカウントで、インストール完了後にアカウントのパスワードを変更する必要があります。

  _PUBLIC アカウントには既定のパスワードはありません。このアカウントが有効になることはないので、パスワードを指定しないでください。

- 通常セキュリティ – 作成されたすべてのアカウントのうち、_PUBLIC を除くアカウントで、特権ユーザ・アカウント用に選択されたものと同じパスワードを受け取ります。インストール完了後にこれらのパスワードを変更し、アカウントごとに独自のパスワードとすることをお勧めします。

  _PUBLIC アカウントには既定のパスワードはありません。このアカウントが有効になることはないので、パスワードを指定しないでください。

- ロック・ダウン・セキュリティ – 作成されたすべてのアカウントのうち、_PUBLIC を除くアカウントで、特権ユーザ・アカウント用に選択されたものと同じパスワードを受け取ります。インストール完了後にこれらのパスワードを変更し、アカウントごとに独自のパスワードとすることをお勧めします。

  _PUBLIC アカウントには既定のパスワードはありません。このアカウントが有効になることはないので、パスワードを指定しないでください。ロック・ダウン・インストールでは、_SYSTEM アカウントも無効になっています。

注意　特に最小のセキュリティによるインストールでは、既定のパスワードはセキュリティ面で脆弱です。この問題を解決するには、そのアカウントを無効にするか、パスワードを変更します。普通はアカウントを無効にすることをお勧めします。

これは特に、コンテナ化されたインスタンスでは重要な問題です。問題を解決する方法を含む詳細は、″認証とパスワード″ を参照してください。

初期のサービス・プロパティ

サービスは、ユーザとコンピュータが InterSystems IRIS に接続するための主要な手段です。インターシステムズのサービスの詳細は、″サービス″ を参照してください。

サービス・プロパティ	最小	通常	ロック・ダウン	説明

サービス・プロパティ	最小	通常	ロック・ダウン	説明
Use 許可が Public	可	可	不可	サービス・リソースに対する Use 許可が Public である場合、あらゆるユーザがそのサービスを利用できます。それ以外の場合は、権限を与えられているユーザのみがそのサービスを利用できます。
認証が必要	不可	可	可	初期設定が通常またはロック・ダウンであるインストールの場合、すべてのサービスで何らかの認証が必要になります (インスタンス認証、オペレーティング・システム・ベース、または Kerberos)。それ以外の場合は、非認証の接続が許可されます。

サービス・プロパティ	最小	通常	ロック・ダウン	説明
有効化されるサービス	最も多い	一部	最も少ない	インストールの初期セキュリティ設定によって、InterSys-tems IRIS を最初に起動したとき、どのサービスが有効になり、どのサービスが無効になるかが決定します。次の表 "有効化されるサービス" は、これらの初期設定を示しています。

テーブル A-1: 有効化されるサービス

サービス	最小	通常	ロック・ダウン
%Service_Bindings	有効	有効	無効
%Service_CacheDirect	有効	無効	無効
%Service_CallIn	有効	無効	無効
%Service_ComPort	無効	無効	無効
%Service_Console*	有効	有効	有効
%Service_ECP	無効	無効	無効
%Service_Monitor	無効	無効	無効
%Service_Telnet*	無効	無効	無効
%Service_Terminal†	有効	有効	有効
%Service_WebGateway	有効	有効	有効

* Windows サーバのみで使用できるサービス

† 非 Windows サーバのみで使用できるサービス

ユーザ・アカウントの構成

インストール・プロセス中に、InterSystems IRIS プロセスをインスタンス所有者として実行するアカウントを選択する必要があります。インストールにより、インスタンス所有者に対して %All ロールを持つ InterSystems IRIS アカウントが作成され、そのアカウントに InterSystems IRIS への完全な管理者アクセスが付与されます。

インスタンス所有者が必要な特権を確実に持つようにするために、新しいユーザ・アカウントを作成しなければならないこともあります。以下のセクションには、必要なアカウントと特権に関する OS 固有の詳細が示されています。

- Windows – "インストール・ガイド" の "Microsoft Windows への InterSystems IRIS のインストール" の章の "Windows ユーザ・アカウント"
- Unix® および Linux – "インストール・ガイド" の "InterSystems IRIS の UNIX®、Linux、および macOS へのインストール" の章の "所有者およびグループの決定"

Kerberos を使用したセキュリティ環境の準備

InterSystems IRIS でサポートされるすべてのプラットフォームには、ベンダが提供およびサポートするバージョンの Kerberos が備わっています。Kerberos を使用するには、Kerberos KDC (Key Distribution Center) または Windows ドメイン・コントローラがネットワークに接続されている必要があります。それぞれをインストールするための準備は、以下のとおりです。

- Windows ドメイン・コントローラ

  この構成では、InterSystems IRIS サーバとクライアントを Windows および非 Windows マシン上で実行し、Windows ドメイン・コントローラを使用して KDC 機能を実現します。ドメイン管理者は、InterSystems IRIS サーバ上で InterSystems サービスを実行するためのドメイン・アカウントを作成します。Windows および非 Windows の InterSystems IRIS サーバを使用する場合の要件については、以下のセクションを参照してください。

  - Windows サーバ用の Windows サービス・アカウントの作成
  - システムで使用するアプリケーションによっては、"Windows Kerberos クライアントの構成" で説明されている手順も実行する必要があります。
  - 非 Windows サーバ用の Windows サービス・アカウントの作成

- 非 Windows の KDC

  この構成では、InterSystems IRIS サーバとすべてのクライアントを非 Windows マシン上で実行し、UNIX® または Kerberos の KDC を実現します。UNIX® または macOS KDC と InterSystems IRIS サーバを使用するための要件については、以下の 2 つのセクションを参照してください。

  - KDC での非 Windows サーバのサービス・プリンシパルの作成
  - Kerberos KDC 機能のテスト

用語に関するメモ

このドキュメントでは、以下の関連する個別のエンティティについて言及します。

- サービス・アカウント – オペレーティング・システム (Window など) 内のエンティティ。ソフトウェア・アプリケーションまたはサービスを表します。
- サービス・プリンシパル – Kerberos のエンティティ。ソフトウェア・アプリケーションまたはサービスを表します。

Windows サーバ用の Windows サービス・アカウントの作成

KDC と、ドメイン・コントローラ上で動作するその他のセキュリティ・サービスを統合することによって、Kerberos 認証プロトコルを実装します。InterSystems IRIS を Windows ドメインにインストールする前に、Windows ドメイン・コントローラを使用して、Windows マシン上の各 InterSystems IRIS サーバ・インスタンスのサービス・アカウントを作成する必要があります。

アカウントの特性

Windows ドメイン・コントローラにこのアカウントを作成する場合は、次のように設定します。

- アカウントの **[パスワードを無期限にする]** プロパティを設定します。

- 作成したアカウントを、InterSystems IRIS サーバ・マシン上の Administrators グループのメンバにします。
- このアカウントを [**サービスとしてログオン**] ポリシーに追加します。

重要　ドメイン全体のポリシーが有効な場合、正常に InterSystems IRIS を機能させるには、このサービス・アカウントをポリシーに追加する必要があります。

名前および名前付け規約

クライアントとサーバが排他的に Windows 上に存在する環境でサービス・プリンシパルに名前を付ける方法には次の 2 種類があります。標準の Kerberos 名前付け規約に従うことも、一意の文字列を使用することもできます。Kerberos 名前付け規約に従うと、今後 Windows 以外のシステムを使用することになった場合にも互換性が保証されます。名前の作成方法の選択によって、サーバへの接続を設定するための手順は多少異なります。

- Kerberos 規約に従った名前では、以下の手順に従います。
  1. Windows の setspn コマンドを実行し、service_principal/fully_qualified_domain_name の形式でサービス・プリンシパルの名前を指定します。service_principal には、通常 `iris` を指定し、fully_qualified_domain_name には、マシン名をドメイン付きで指定します。例えば、`iris/irisserver.example.com` のようにサービス・プリンシパル名を指定します。setspn ツールの詳細は、Microsoft のドキュメントの "Setspn" のページを参照してください。
  2. [InterSystems IRIS サーバ・マネージャ] ダイアログで、新しい優先サーバを追加するために [Kerberos] を選択します。[**サービスプリンシパル名**] フィールドに指定する名前は setspn で指定したプリンシパル名と一致する必要があります。
- 任意の一意の文字列を使った名前では、以下の手順に従います。
  1. サービス・プリンシパルの名前を選択します。InterSystems IRIS サーバ・インスタンスを表す各アカウントについて推奨される名前付け規約は "`iris`HOST" の形式で、リテラル `iris` に続けてホスト・コンピュータ名を大文字で指定します。例えば、WINSRVR という Windows マシン上で InterSystems IRIS サーバを実行する場合、ドメイン・アカウント名は `irisWINSRVR` となります。
  2. [InterSystems IRIS サーバ・マネージャ] ダイアログで、新しい優先サーバを追加するために [Kerberos] を選択します。[**サービスプリンシパル名**] フィールドに、サービス・プリンシパルに対して選択した名前を指定します。

リモート・サーバ接続の構成手順の詳細は、"リモート・サーバへの接続" を参照してください。

Windows Kerberos クライアントの構成

Windows クライアントで Kerberos を使用している場合、ユーザに認証情報の入力を求めないよう、クライアントの構成が必要になることもあります。これは、資格情報を要求できないプログラムを使用している場合に必要となります。そうでないと、プログラムは接続不可能となります。

資格情報の入力を求めないように Windows を構成するには、以下の手順を実行します。

1. Windows クライアント・マシンで、レジストリ・エディタ regedit.exe を起動します。
2. HKEY_LOCAL_MACHINE¥System¥CurrentControlSet¥Control¥Lsa¥Kerberos¥Parameters キーに進みます。
3. このキーで、AllowTgtSessionKey の値を 1 に設定します。

非 Windows サーバ用の Windows サービス・アカウントの作成

InterSystems IRIS を Windows ドメインにインストールする前に、Windows ドメイン・コントローラを使用して、非 Windows マシン上の各 InterSystems IRIS サーバ・インスタンスのサービス・アカウントを作成する必要があります。そのマシン上に存在する InterSystems IRIS サーバ・インスタンスの数にかかわらず、マシンごとにサービス・アカウントを 1 つ作成してください。

通常、これらのアカウントには“irisHOST”という名前を付けます。irisという文字列の後に、ホスト・コンピュータ名を大文字で指定してください。例えば、UNIXSRVRという非 Windows マシン上で InterSystems IRIS サーバを実行する場合、ドメイン・アカウント名は irisUNIXSRVR となります。非 Windows プラットフォーム上の InterSystems IRIS サーバの場合、このアカウントが Kerberos サービス・プリンシパルにマップされます。

重要　Windows ドメイン・コントローラでこのアカウントを作成する際、InterSystems IRIS では、そのアカウントに[パスワードを無期限にする]のプロパティを設定する必要があります。

Windows ドメインで非 Windows InterSystems IRIS サーバを設定するには、Windows ドメインから keytab ファイルを取得する必要があります。keytab ファイルは、InterSystems IRIS サーバのサーバ名とそのキーが格納されているファイルです。

そのためには、Windows サービス・アカウント(この例では irisUNIXSRVR)を InterSystems IRIS サーバ上のサービス・プリンシパルにマップし、ドメイン・コントローラで ktpass コマンドライン・ツールを使用して、アカウントからキーを取得します。これは、Windows サポート・ツールのひとつとして Microsoft が提供しています。

このコマンドを実行すると、設定したアカウントが UNIX®/Linux マシン上のアカウントにマップされます。さらに、そのアカウントのキーも生成されます。このコマンドでは以下のパラメータを指定する必要があります。

パラメータ	説明
/princ	iris/<fully qualified hostname>@<kerberos realm> の形式で表されたプリンシパル名。
/mapuser	作成されたアカウントの名前 (iris<HOST> の形式)。
/pass	アカウントの作成時に指定されたパスワード。
/crypto	使用する暗号化の種類。指定しない場合は、既定値が使用されます。
/out	生成する keytab ファイル。InterSystems IRIS サーバ・マシンに渡し、既存の keytab ファイルと交換またはマージします。

重要　UNIX®/Linux プラットフォームのプリンシパル名はこのテーブルに示す形式 (リテラル iris を先頭部分に置きます) で指定する必要があります。

キー・ファイルを生成した後、InterSystems IRIS サーバ上のファイルにそのキー・ファイルを移動します。キー・ファイルの特徴は以下のセクションで説明されています。

KDC での非 Windows サーバのサービス・プリンシパルの作成

非 Windows 環境では、UNIX®/Linux または macOS の KDC を使用する UNIX®/Linux または macOS の各 InterSystems IRIS サーバにサービス・プリンシパルを作成する必要があります。サービス・プリンシパル名は、iris/<fully qualified hostname>@<kerberos realm> の形式で表されます。

キー・ファイルの特徴

このプリンシパルを作成したら、InterSystems IRIS サーバ上のキー・ファイルにそのキーを抽出します。キー・ファイルの特徴を以下に示します。

- ほとんどのバージョンの UNIX® でのパス名は install-dir/mgr/iris.keytab です。macOS および SUSE Linux でのパス名は /etc/krb5.keytab です。
- このファイルは、InterSystems IRIS インストールを所有するユーザおよびグループ irisusr が所有します。
- このファイルのアクセス許可は、640 です。

Kerberos KDC 機能のテスト

非 Windows のサーバとクライアントのみで構成したシステムで Kerberos を使用する場合は、Windows のドメイン・コントローラを使用するより、UNIX®/Linux のネイティブな KDC を使用した方が簡単です。KDC のインストール方法と構成方法については、各ベンダのドキュメントを参照してください。通常、KDC のインストールと構成はシステム管理者が行います。

Kerberos をインストールするときは、以下の 2 つのソフトウェア・セットをインストールします。

- KDC。Kerberos サーバ・マシンにインストールします。
- クライアント・ソフトウェア。Kerberos クライアントをホストするすべてのマシンにインストールします。このソフトウェア・セットは、オペレーティング・システムによって大きく異なる場合があります。クライアント・ソフトウェアの種類とそのインストール方法については、オペレーティング・システム・ベンダのドキュメントを参照してください。

必要な Kerberos ソフトウェアをインストールしたら、kadmin、kinit、および klist コマンドを使用した簡単なテストを実行できます。これらのコマンドは、ユーザのプリンシパルを Kerberos データベースに追加し、ユーザの TGT (Ticket-Granting Ticket) を取得したうえで、その TGT を列記します。

テストが完了し、登録されたプリンシパルのチケットを Kerberos が提供できることを確認したら、InterSystems IRIS をインストールする準備は完了です。

システム管理およびセキュリティ

システム管理およびセキュリティ

このページでは、InterSystems IRIS® データ・プラットフォームの管理ポータルへのアクセス、およびポータルのその他のセキュリティ関連機能について説明します。

InterSystems IRIS のセキュリティ・ドメインの管理

インターシステムズのセキュリティ・ドメインでは、Kerberos レルムと Windows ドメインに対応するグループにユーザが分類されます。Kerberos を使用しているインスタンスの場合、その InterSystems IRIS ドメインは Kerberos レルムに対応しています。Windows ドメインを使用している場合は、そのドメインが Kerberos レルムに対応します。

セキュリティ・ドメインの名前は、インターネット・ドメインの名前と同じ形式をとることが普通ですが、必ずしもこれは必須ではありません。セキュリティ・ドメイン名には、アット・マーク (@) 以外のあらゆる文字を使用できます。

単一ドメインと複数ドメイン

InterSystems IRIS は、単一ドメインまたは複数ドメインの使用をサポートしています。

単一ドメインまたは複数ドメインのサポートを指定するには、"システム規模のセキュリティ・パラメータ" で説明されている、管理ポータルの **[システムワイドセキュリティパラメータ]** ページ (**[システム管理]→[セキュリティ]→[システム・セキュリティ]→[システムワイドセキュリティパラメータ]**) の **[複数セキュリティドメインを許可する]** フィールドを使用します。

単一ドメインを持つインスタンスでは、以下のようになります。

- $USERNAME 変数にはドメイン名は含まれていません。
- システム・ユーティリティでユーザ名を表示すると、ドメイン名は表示されません。
- 既定のドメイン以外のドメインにあるユーザ名を指定することはできません (以下のセクションで説明します)。

複数ドメインを持つインスタンスでは、以下のようになります。

- $USERNAME 変数にはドメイン名が含まれています。
- システム・ユーティリティでユーザ名を表示すると、ドメイン名も表示されます。これには、**[ ]** ページ (**[セキュリティ管理]** > **[セキュリティ]** > **[ユーザ]**) も含まれます。
- ユーザは、各自のドメインにおける完全修飾名 (documentation@intersystems.com など) を使用してログインします。完全修飾名の最初の部分を共有し、ドメイン名が異なる 2 つのアカウントがある場合、これらは 2 つの別個のユーザ・アカウントとして格納されます (それぞれに独自の属性があり、これらの属性に異なる値を指定できます)。
- ユーザ名を編集することはできません。

既定のセキュリティ・ドメイン

インスタンスごとに既定のセキュリティ・ドメインがあります。ユーザ名にドメインが指定されていない場合には、既定のドメイン名が使用されていると見なされます。例えば、既定のドメインが “intersystems.com” である場合、“info” と “info@intersystems.com” は同じものを指します。InterSystems IRIS をインストールすると、このパラメータの初期値にはローカル・ドメイン名が使用されます。

複数のセキュリティ・ドメインを持つインスタンスでは、"システム規模のセキュリティ・パラメータ" のセクションで説明されているように、**[システムワイドセキュリティパラメータ]** ページ (**[システム管理]→[セキュリティ]→[システム・セキュリティ]→[システムワイドセキュリティパラメータ]**) の **[デフォルトセキュリティドメイン]** フィールドを使用して既定のセキュリティ・ドメインを新規に選択できます。

セキュリティ・ドメインのリスト、作成、編集、削除

[LDAP 構成] ページには、インスタンスの既存のセキュリティ・ドメインと構成がリストされ、構成やドメインを作成したり、既存のドメインや構成を変更したり、削除したりできます。

セキュリティ・ドメインのリスト

インスタンスのドメインのリストを表示するには、[セキュリティ LDAP 構成] ページ ([システム管理] > [セキュリティ] > [システム・セキュリティ] > [LDAP 構成]) に移動します。このページには、ドメインごとに以下の情報が表示されます。

- [ログインドメイン名] – ドメインの名前。クリックすると、ドメインのプロパティを編集できます。
- [LDAP 有効] – このドメインで LDAP 接続を有効にするかどうか。
- [説明] – ドメインの説明。
- [削除] リンク – 確認後、ドメインをインスタンスから削除します。

注釈　インスタンスで Kerberos が有効になっている場合、このページに移動するには、メニュー [LDAP/Kerberos 構成] を選択します。ページの名前は [セキュリティ LDAP/Kerberos 構成] です。

セキュリティ・ドメインの作成

使用するインスタンスのドメインを作成するには、そのドメインを指定する LDAP 構成を作成します。LDAP 構成を作成すると、ドメインが作成されます。

1. [セキュリティ LDAP 構成] ページ ([システム管理] > [セキュリティ] > [システム・セキュリティ] > [LDAP 構成]) に移動します。
2. [新しい LDAP 構成を作成] ボタンをクリックします。このボタンを選択すると、[LDAP 構成を編集] ページが表示されます。
3. [LDAP 構成を編集] ページで、[ログインドメイン名] とオプションの説明を入力します。
4. 他の構成フィールドに値を入力して [保存] をクリックし、構成とドメインを作成します。

注釈　インスタンスで Kerberos が有効になっている場合、このページに移動するには、メニュー [LDAP/Kerberos 構成] を選択します。ページの名前は [セキュリティ LDAP/Kerberos 構成] です。

セキュリティ・ドメインの編集

ドメインを編集するには、以下の手順を実行します。

1. [セキュリティ LDAP 構成] ページ ([システム管理] > [セキュリティ] > [システム・セキュリティ] > [LDAP 構成]) に移動します。
2. [ログインドメイン名] をクリックして、ドメインとその構成フィールドを編集します。
3. [保存] をクリックして、変更した構成とドメインを保存します。

注釈
1. ドメインの名前は変更できません。または、目的の名前のドメインが必要な場合は、その名前でドメインを新規に作成した後で既存のドメインを削除します。
2. インスタンスで Kerberos が有効になっている場合、このページに移動するには、メニュー [LDAP/Kerberos 構成] を選択します。ページの名前は [セキュリティ LDAP/Kerberos 構成] です。

セキュリティ・ドメインの削除

ドメインを削除するには、以下の手順を実行します。

1. [セキュリティ LDAP 構成] ページ ([システム管理] > [セキュリティ] > [システム・セキュリティ] > [LDAP 構成]) に移動します。ドメインのリストが表示されます。
2. ドメインの行で [削除] をクリックします。
3. 削除を確認します。

注釈　インスタンスで Kerberos が有効になっている場合、このページに移動するには、メニュー [LDAP/Kerberos 構成] を選択します。ページの名前は [セキュリティ LDAP/Kerberos 構成] です。

パスワードの強固さとパスワードのポリシー

InterSystems IRIS では、以下の形式の文字列を使用することでユーザ・パスワードに対する要件を指定できます。

```
X.Y[ANP]
```

以下はその説明です。

- X は、パスワードに使用する文字の最小数です。
- Y は、パスワードに使用する文字の最大数です。
- A、N、および P は、それぞれアルファベット文字、数字、および句読点文字をパスワードに使用できるかどうかを指定します。

これらの規則は、ObjectScript のパターン・マッチング機能に基づいています。この機能は、"パターン・マッチング" で説明されています。

注釈　このパラメータの設定によって、既存のパスワードが影響を受けることはありません。

管理者パスワードに推奨される強固さ

管理者パスワードは、大文字と小文字のアルファベット文字、数字、および句読点文字が無作為に混在したものとすることが理想的です。このようなランダムな文字を使用して、最低でも 12 文字のパスワードを使用することを強くお勧めします。

緊急アクセス

InterSystems IRIS には、特定の緊迫した状況下で使用できる特別な緊急アクセス・モードが用意されています。このような状況としては、セキュリティ構成情報に深刻な損傷が発生した場合、**%Admin_Manage:Use** 特権または **%Admin_Security:Use** 特権を持っているユーザが存在しない場合 (つまり、すべてのユーザがロックアウトされている場合) などがあります。InterSystems IRIS には、このような事態を回避するために **%All** ロールを付与されたユーザを常に 1 人以上確保する仕組みがありますが、そのユーザが不在であることや、そのユーザがパスワードを忘れてしまうこともあります。

InterSystems IRIS のインスタンスへの緊急アクセスを取得するには、インスタンスが実行されている場所で root 特権または管理者特権 (インスタンスを root でインストールした場合) を持っているか、インスタンスをインストールしたユーザである必要があります (インスタンスを root でインストールしていない場合)。このような要件があるため、緊急アクセスは、インスタンスに対する管理操作 (新しいインスタンスを既存のインスタンス上にインストールするなど) を実行するための十分な特権を既に持っているユーザに限られます。

緊急アクセスに関するトピック:

- 緊急アクセスの仕組み
- Windows での緊急アクセス・モードの呼び出し
- UNIX®、Linux、および macOS での緊急アクセス・モードの呼び出し

緊急アクセス・モードの仕組み

InterSystems IRIS が緊急アクセス・モードで動作しているとき、アクセスが許可されるユーザは 1 人のみです (このユーザを緊急ユーザといいます)。このユーザ名は、InterSystems IRIS 内であらかじめ定義されていなくてもかまいません。インスタンスに既に緊急ユーザと同じ名前を持つユーザ・アカウントがある場合、緊急ユーザは、既存の標準ユーザ・アカウントの特権ではなく、緊急アクセス・モードに関連する特権を持ちます。

緊急ユーザのアカウントとパスワードは、緊急モードでの 1 回の起動のみに対して有効です。緊急ユーザに指定したユーザ名が、InterSystems IRIS のインスタンスで以前に定義したユーザ名である場合、システムを通常モードで再起動すると、以前に定義したそのユーザの元のパスワードとセキュリティ特権が復元されます。緊急ユーザに指定したユーザ名が新規である場合、InterSystems IRIS を通常モードで再起動するときに、その新規ユーザのアカウントが無効であっても、そのユーザのログイン資格情報とセキュリティ特権が保存されます。

Tipヒント **%ALL** ロールが登録された休眠アカウントが InterSystems IRIS のインスタンスに蓄積されないように、緊急ユーザには、新規ユーザ名ではなく以前に定義したユーザ名を使用することをお勧めします。これにより、複数の管理者がいるシステムでは、それぞれの管理者がそのユーザ名を使用して緊急アクセス・モードを初期化するのであれば、緊急アクセス・モード中に発生した変更の作成情報を、ログを通じて追跡することもできます。

緊急アクセス・モードの InterSystems IRIS には、以下の制約と機能があります。

- 緊急ユーザのみにアクセスが許可されます。他のユーザはログインできません。緊急ユーザは **%ALL** ロールを保持します。
- インスタンス認証を使用するアクセスのみになります。他の認証メカニズムはサポートされません。2 要素認証は無効になります。これによって、2 要素認証で緊急ユーザが認証できない状況が回避されます。
- Web アプリケーションでポータル (/csp/sys および /csp/sys/*) を制御する場合、カスタム・ログイン・ページが利用できても、緊急アクセス中には標準のログイン・ページ (%CSP.Login.cls) が使用されます。カスタム・ログイン・ページでは認証が行われなくなる場合があるので、これによって緊急ユーザが確実にポータルにアクセスできるようにします。他の Web アプリケーションの場合、カスタム・ログイン・ページがあると、そのページが緊急ログインの際に使用されます。
- 緊急アクセスによるログイン後、InterSystems IRIS は、アクティブ・プロセスに対するすべてのイベントの監査を試みます。これが不可能であっても、InterSystems IRIS の起動は継続します。緊急アクセス・モードでのログインの失敗は、監査されません。
- 有効になっているサービスは、コンソール、ターミナル、および Web ゲートウェイ (**%Service_Console**、**%Service_Terminal**、および **%Service_WebGateway**) のみです。それ以外のサービスはすべて無効化されます。これによって、緊急モード以外のモードで InterSystems IRIS が起動するときに、サービスの有効または無効の状態が変化することはありません。影響を受けるのは、現時点 (緊急モード) でメモリにある、サービスに関する情報のみです。
- 有効なサービスに対しては、認証されたアクセスのみが許可されます。該当のサービスに対しては InterSystems IRIS 独自のパスワード認証が使用されるので、緊急ユーザのユーザ名とパスワードを使用する必要があります。
- 緊急ユーザは InterSystems IRIS の構成を変更できますが、その変更が有効になるのは、次回に InterSystems IRIS を通常のモードで起動したときです。緊急モードで起動しても有効になりません。これは、InterSystems IRIS の通常の動作とは異なる点です。通常の動作では、InterSystems IRIS を再起動しなくても、構成の変更はほとんどその場で有効になります。

Windows での緊急アクセス・モードの呼び出し

InterSystems IRIS を緊急アクセス・モードで起動するには、ユーザは管理者グループのメンバである必要があります。以下の手順を実行します。

1. コマンド・プロンプトを起動して、管理者として実行します。以下のいずれかの方法で行います。

- Windows コマンド・プロンプト・プログラム。メニューの **[コマンド プロンプト]** 選択項目を右クリックして、**[管理者として実行]** を選択します。
- Windows PowerShell。管理者または他の特権のないユーザとしてこれを実行できますが、この手順では管理者として実行していることを想定しています。他の特権のないユーザとして実行するには、コマンドを呼び出すときに `-verb runas` 引数を使用します。PowerShell のドキュメントを参照してください。

2. InterSystems IRIS のインストール先にある **bin** ディレクトリに移動します。
3. このディレクトリで、コマンド行から適切なスイッチを使用し、緊急ユーザのユーザ名とパスワードを指定して、InterSystems IRIS を呼び出します。これは、使用しているコマンド・プロンプトによって決まります。

   - Windows コマンド・プロンプトの場合のコマンドは以下のとおりです。

     ```
     iris start <instance> /EmergencyId=<username>,<password>
     ```

     これにより、ユーザが 1 人だけ認められる緊急モードの InterSystems IRIS セッションが開始します。各パラメータは以下のとおりです。

     - 〈instance〉 は、緊急モードで開始するインスタンスを指定します。
     - 〈username〉 は、システムの唯一のユーザです。
     - 〈password〉 は、そのユーザのパスワードです。

   - Windows PowerShell の場合のコマンドは以下のとおりです。

     ```
     start-process .\iris.exe -ArgumentList "start <instance> /EmergencyId=<username>,<password>"
     ```

     これにより、ユーザが 1 人だけ認められる緊急モードの InterSystems IRIS セッションが開始します。各パラメータは以下のとおりです。

     - 〈instance〉 は、緊急モードで開始するインスタンスを指定します。
     - 〈username〉 は、システムの唯一のユーザです。
     - 〈password〉 は、そのユーザのパスワードです。

   注釈　Windows では、他のオペレーティング・システムと異なり、`EmergencyId` スイッチの前にスラッシュ (“/”) を記述します。

   例えば、MyIRIS というインスタンスで、purple22 というパスワードを持つユーザ jmd が InterSystems IRIS を緊急モードで起動する場合は、以下のように入力します。

   ```
   iris start MyIRIS /EmergencyId=jmd,purple22
   ```

   この状態でログインできるのは、以下のように適切なパスワードを使用する緊急ユーザのみです。

   ```
   Username: jmd
   Password: ********
   Warning, bypassing system security, running with elevated privileges
   ```

InterSystems IRIS が起動すると、InterSystems IRIS ランチャーからターミナルを起動できるほか、任意の Web アプリケーションを実行することもできます。これによって、管理ポータルおよび文字ベースのすべてのユーティリティにアクセスできるようになります。これらにアクセスし、必要に応じて設定を変更した後、InterSystems IRIS を通常のモードで再起動します。

UNIX®、Linux、および macOS での緊急アクセス・モードの呼び出し

InterSystems IRIS を緊急アクセス・モードで起動するには、root アクセス権を持っているか、対象インスタンスの所有者である必要があります。コマンド行から適切なスイッチを使用し、緊急ユーザのユーザ名とパスワードを指定して、InterSystems IRIS を呼び出します。

```
./iris start <instance-name> EmergencyId=<username>,<password>
```

これにより、ユーザが 1 人だけ認められる緊急モードの InterSystems IRIS セッションが開始します。各パラメータは以下のとおりです。

- 〈instance-name〉 は、緊急モードで開始するインスタンスを指定します。
- 〈username〉 は、システムの唯一のユーザです。
- 〈password〉 は、〈username〉 のパスワードです。

注釈　これらのオペレーティング・システムのいずれかから Windows に移行する場合、Windows では `EmergencyId` スイッチの前にスラッシュ (“/”) が必要になる点に注意します。

例えば、MyIRIS というインスタンスで、purple22 というパスワードを持つユーザ jmd が InterSystems IRIS を緊急モードで起動する場合は、以下のように入力します。

```
./iris start MyIRIS EmergencyId=jmd,purple22
```

この状態でログインできるのは、以下のように適切なパスワードを使用する緊急ユーザのみです。

```
Username: jmd
Password: ********
Warning, bypassing system security, running with elevated privileges
```

InterSystems IRIS が起動すると、ターミナルまたは任意の Web アプリケーションを実行できます。これによって、管理ポータルおよび文字ベースのすべてのユーティリティにアクセスできるようになります。これらにアクセスし、必要に応じて設定を変更した後、InterSystems IRIS を通常のモードで再起動します。

[システムセキュリティ設定] ページ

[システムセキュリティ設定] ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]**) には、InterSystems IRIS® インスタンス全体のセキュリティを構成するページへのリンクが用意されています。このページは以下のとおりです。

- システム規模のセキュリティ・パラメータ
- 認証/Web セッション・オプション
- LDAP オプション

システム規模のセキュリティ・パラメータ

ここでは、InterSystems IRIS のインスタンス全体に影響するセキュリティの問題について説明します。システムワイド・セキュリティ・パラメータや、メモリ・イメージに存在する機密性の高いデータの扱いなどを取り上げます。

InterSystems IRIS には、多数のシステム規模のセキュリティ・パラメータが用意されています。これらのパラメータは、**[システムセキュリティ設定]** ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[システムワイドセキュリティパラメータ]**) で構成できます。以下のとおりです。

- **[監査を有効に]** ー 監査を有効または無効にします。このチェック・ボックスの機能は、**[監査]** ページ (**[システム管理]** > **[セキュリティ]** > **[監査]**) にある **[監査を有効に]** リンクおよび **[監査を無効に]** リンクと同じです。監査の詳細は、″監査ガイド″ を参照してください。既定では無効になっています。

- **[監査データベースにエラーが発生したときにシステムをフリーズしますか]** – (監査が有効な場合にのみ使用可能)。監査データベースへの書き込み中にエラーが発生した場合、インスタンスを停止 (フリーズ) します。詳細は、"監査データベースに書き込めない場合のシステムのフリーズ" を参照してください。
- **[構成セキュリティを有効に]** – "構成のセキュリティ" の説明にあるように、構成のセキュリティを有効にするかどうかを指定します。既定では無効になっています。
- **[デフォルトセキュリティドメイン]** – インスタンスの既定のセキュリティ・ドメインを選択できます。セキュリティ・ドメインの詳細は、"InterSystems IRIS のセキュリティ・ドメインの管理" を参照してください。既定のドメインは、インストールのときに設定されたドメインです。
- **[不活動上限 (0–365)]** – ユーザ・アカウントを使用しない状態で保持できる最大日数を指定します。これは、2 回の正常なログインの間の時間間隔です。アカウントを使用しない期間がこの限度に達すると、そのアカウントは無効になります。0 を指定しておくと、アカウントを使用せずに放置しても、そのアカウントは無効になりません。既定値は、"初期のユーザ・セキュリティ設定" に説明があります。

  注釈　ミラー・メンバでは、InactiveLimit パラメータは起動時に自動的に 0 に設定されます。これにより、ユーザ・アカウントが他のミラー・メンバ上で非アクティブになるのを防ぎます。

- **[不正ログイン回数制限 (0 – 64)]** – 連続して失敗できるログインの最大回数を指定します。ログインの失敗回数がこの限度に達すると、アカウントが無効になるか、ログインしようとするたびに遅延時間が長くなります。このアクションは、**[ログイン制限に達するとアカウントを無効にする]** フィールドの値で決まります。0 を指定しておくと、何回ログインに失敗しても、引き続き通常どおりにログインを試すことができます。既定値は 5 です。
- **[ログイン制限に達するとアカウントを無効にする]** – チェックを付けておいた場合は、不正ログインの回数が、前に説明したフィールドで指定した回数に達すると、そのユーザ・アカウントは無効になります。
- **[パスワード有効期限 (0–99999)]** – パスワードが無効になる頻度、つまりパスワードの変更が必要になる頻度を日数で指定します。最初に設定するときは、パスワードが無効になるまでの日数を指定します。0 を指定すると、パスワードはいつまでも有効です。ただし、このフィールドを 0 に設定しても、**[次回ログイン時にパスワード変更]** フィールドが設定されているユーザには影響しません。既定値は 0 です。

  注意　この設定は、InterSystems IRIS 自身で使用されているアカウントも含め、該当の InterSystems IRIS インスタンスのすべてのアカウントに影響します。これらのアカウントでは、パスワードを更新しないとさまざまな処理が実行できず、予測できない結果となることがあります。

- **[パスワードパターン]** – 新規に作成するパスワードで使用できる形式を指定します。詳細は、"パスワードの強固さとパスワードのポリシー" を参照してください。既定値は、"初期のユーザ・セキュリティ設定" に説明があります。
- **[パスワード検証ルーチン]** – パスワードを検証するためにユーザによって提供されたルーチン (またはエントリ・ポイント) を指定します。詳細は、Security.System クラスの PasswordValidationRoutine メソッドを参照してください。
- **[このシステムへの接続に必要なロール]** – 既存のロールに設定すると、ユーザがシステムにログインするためには、そのユーザは、ログイン・ロールとしてのその既存のロールのメンバであることが必要になります。

  LDAP 認証または OS ベースの LDAP 認証を使用している場合は、接続に必要なロールを作成して、このフィールドにそのロール名を指定することを強くお勧めします。詳細は、"ログインに必要なロールの設定" を参照してください。

- **[パーセントで始まるグローバルへの書き込みを有効に]** – パーセントで始まるグローバルに対する書き込みアクセス権を暗黙的にすべてのユーザに与えるかどうかを指定します。チェックを外すと、書き込みアクセス権は通常のセキュリティ機能で制御されます。パーセントで始まるグローバルおよびそれを保持しているデータベースである IRISSYS の詳細は、"IRISSYS (マネージャ・データベース)" を参照してください。既定の設定では、アクセス権は通常のセキュリティ機能で制御されます。
- **[複数セキュリティドメインを許可する]** – 複数のインターシステムズのセキュリティ・ドメインをサポートするかどうかを指定します。セキュリティ・ドメインの詳細は、"InterSystems IRIS のセキュリティ・ドメインの管理" を参照してください。既定では、単一のドメインのみがサポートされています。

- **[Telnetサーバ SSL/TLS サポート]** ― Telnet サーバがクライアント接続に TLS の使用をサポートまたは要求するかを指定します。

  重要　　TLS を使用するよう Telnet サーバを構成する前に、**%TELNET/SSL** と呼ばれる構成が存在する必要があります。InterSystems IRIS Telnet サーバでの TLS の使用に関する詳細は、“TLS を使用するための InterSystems IRIS Telnet サーバの構成”を参照してください。

  オプションは以下のとおりです。

  - **[無効]** ― TLS を使用するクライアント接続を拒否します(つまり、TLS を使用しないクライアント接続のみを受け入れます)。
  - **[有効]** ― TLS を使用するクライアント接続を受け入れますが、それは必須ではありません。
  - **[必須]** ― TLS を使用するクライアント接続を要求します。

- **[デフォルトの署名ハッシュ]** ― XML 署名ハッシュを作成するために既定で使用されるアルゴリズムを指定します。ハッシュの作成でサポートされるアルゴリズムの詳細は、https://www.w3.org/ を参照してください。

認証オプション

[認証/Web セッション・オプション] ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[認証/Web オプション]**) では、InterSystems IRIS インスタンス全体の認証メカニズムを有効または無効にできます。

- InterSystems IRIS インスタンス全体で無効に設定した認証メカニズムは、どのサービスでも使用できなくなります。
- InterSystems IRIS インスタンス全体で有効に設定した認証メカニズムは、それをサポートするすべてのサービスで使用できます。特定のサービスに対して認証メカニズムを有効にするには、その該当のプロパティの **[サービス編集]** ページを使用します。このページは、**[サービス]** ページ (**[システム管理]** > **[セキュリティ]** > **[サービス]**) からサービスを選択することで利用できます。

  注釈　　すべてのサービスがすべてのメカニズムをサポートするわけではありません。

認証オプションは以下のとおりです。

- **[認証なしアクセスを許可]** ― ユーザは認証なしに接続できます(ログイン・ダイアログが表示されたら、**[ユーザ名]** および **[パスワード]** フィールドを空白にしたまま **[OK]** をクリックしてログインできます)。
- **[OS 認証を許可]** ― オペレーティング・システムのユーザ ID を使用してユーザを識別してから、インターシステムズの承認を使用します。
- **[代行認証によるOS認証を許可]** ― オペレーティング・システムのユーザ ID を使用してユーザを識別してから、代行承認を使用します。
- **[LDAP認証によるOS認証を許可]** ― オペレーティング・システムのユーザ ID を使用してユーザを識別してから、LDAP 承認を使用します。
- **[パスワード認証を許可]** ― インスタンス認証と呼ばれる InterSystems IRIS 独自のネイティブ・ツールを使用してユーザを認証してから、インターシステムズの承認を使用します。
- **[代行認証を許可]** ― 外部 (代行) 認証システムを呼び出して、使用します。代行認証は、インターシステムズの承認または代行承認と共に使用できます。
- **[Always try Delegated authentication]** ― InterSystems IRIS は、インスタンス認証 (パスワード認証とも呼ばれる) で認証するユーザに対して代行認証コードを呼び出します。代行認証とインスタンス認証の両方を使用し、インスタンス認証ユーザには ZAUTHENTICATE の呼び出しも求める場合は、このオプションを選択します。
- **[Kerberos認証を許可]** ― Kerberos を使用して認証を実行します。Kerberos 認証は、インターシステムズの承認または代行承認と共に使用できます。

- **[LDAP認証を許可]** – LDAP (Active Directory を含む) を使用してユーザを認証します。LDAP を認証と承認の両方に使用することも、LDAP をインターシステムズの承認と共に使用することもできます。
- **[LDAPキャッシュ credentials 認証を許可]** – キャッシュした LDAP 認証情報のコピーを使用して、LDAP データベースを利用できない場合でも LDAP ユーザを認証します。
- **[ログイン Cookie の作成を許可]** – InterSystems IRIS は、有効になっている Web アプリケーション間で共有される Cookie を使用してユーザを認証するため、新しいアプリケーションを初めて使用するときにユーザ名およびパスワードを入力する必要はありません。これは、CSP を使用する Web アプリケーションのみに関連します。
- **[ログイン Cookie の有効期間 (秒)]** – ログイン Cookie の有効期間 (秒)。このフィールドは、インスタンスに対してログイン Cookie が有効になっている場合のみ関係します。
- **[二要素タイムベースのワンタイム・パスワード認証を許可]** – InterSystems IRIS は、認証デバイス、またはユーザの携帯電話上で動作しているアプリケーションを使用して検証コードを提供します。その後ユーザはこのコードを入力して認証プロセスを完了します。これを選択すると、**[認証/ウェブセッションオプション]** ページに 2 要素認証を構成するためのフィールドが表示されます。
- **[二要素 SMS テキスト認証を許可]** – InterSystems IRIS は、携帯電話のテキスト・メッセージを使用してセキュリティ・コードを提供します。その後ユーザはこのコードを入力して認証プロセスを完了します。これを選択すると、**[認証/ウェブセッションオプション]** ページに 2 要素認証を構成するためのフィールドが表示されます。

複数の認証オプションがサポートされている場合は、カスケード認証が使用されます。

変更内容の反映

さまざまなセキュリティ設定を変更した場合、その変更が反映されるまでの時間は以下のとおりです。

- ユーザのプロパティ (割り当てられたロールなど) に対する変更は、そのユーザが次回ログインしたときに有効になります。既に実行中のプロセスには反映されません。
- サービスに対する変更 (サービスを有効にするかどうか、認証を要求するかどうかなど) は、今後そのサービスに接続しようとしたときに有効になります。既存の接続には反映されません。
- ロールの定義に対する変更は、以降行われる特権のチェックですぐに有効になります。これらの変更はデータベースへのアクセスが発生するたびにチェックされるので、データベース・リソースに対しては直ちに反映されます。サービスとアプリケーションに対しては、変更の時点以降にサービスに接続しようとしたとき、またはアプリケーションを起動したときに反映されます。

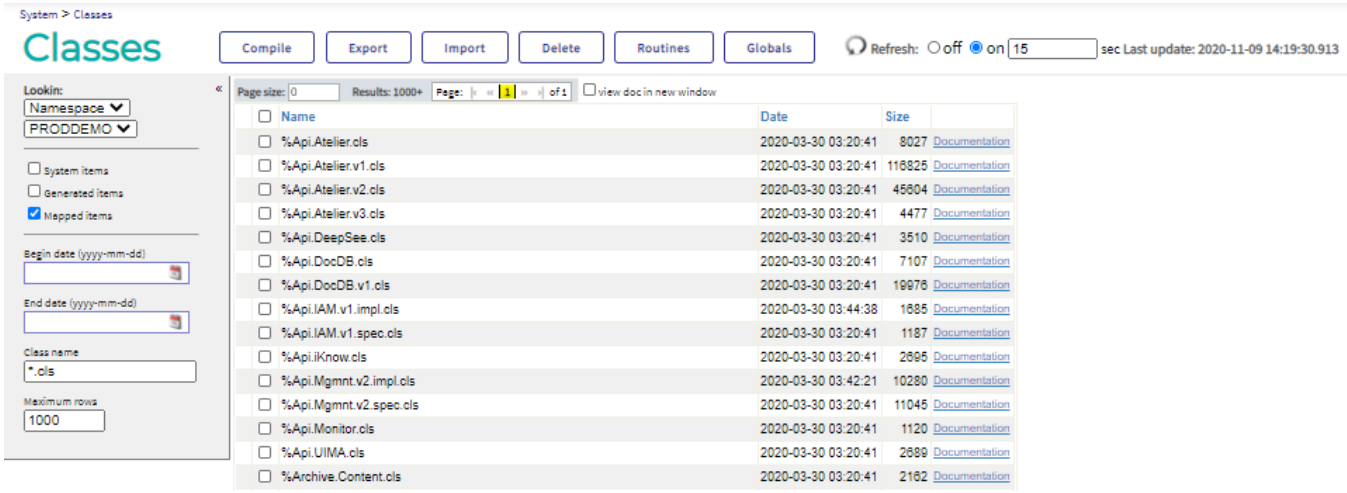
注釈　ここに挙げた時間は、変更が反映されるまでの最長時間なので、もっと早い時点で変更が有効になることもあります。

管理ポータルのページの自動更新の有効化

既定では、ユーザは  (**[ダイアグラムを更新]**) アイコンが使用可能な場合に、これをクリックすることによってのみ、管理ポータルのページを更新できます。ただし、InterSystems IRIS では、数秒ごとに管理ポータルのページを自動更新するメカニズムをユーザに提供できます。次に示すように、ターミナルから、このメカニズムを公開するよう ^%SYS グローバルを変更できます。

```
set ^%SYS("Portal","EnableAutoRefresh") = 1
```

これを行うと、管理ポータルの更新可能なページに一連のラジオ・ボタンが表示され、ユーザが自動更新のオン/オフを切り替えることができるようになります。一部のページでは、ユーザが更新間隔を指定することもできます。重要なこととして、管理者が `EnableAutoRefresh` ノードを `1` に設定すると、自動更新は既定でオフになります。以下の図は、自動更新が有効になっていて、更新間隔が 15 秒に設定されている **[クラス]** ページを示しています。



重要　　自動更新を行うとInterSystems IRIS サーバへの呼び出しが発生するため、自動ログアウトが有効になっている場合、ログアウトできなくなることがあります。詳細は、"管理ポータルの自動ログアウト動作" を参照してください。

管理ポータルの自動ログアウト動作

InterSystems IRIS 管理ポータル Web アプリケーションにはそれぞれ [セッションタイムアウト] プロパティがあり、これによって、ユーザのセッションが期限切れになるまでユーザが非アクティブな状態でいられる時間が指定されています。既定では、ユーザのセッションが期限切れになってから 15 秒後に、管理ポータルは現在のページを更新してユーザをログアウトさせます。その際、保留中の変更はキャッシュされません。また、保留中の変更の保存を求めるプロンプトも表示されません。保存されていない変更は破棄されます。

重要　　非アクティブ状態の時間は、InterSystems IRIS サーバの呼び出しから次の呼び出しまでです。すべてのユーザ・アクションがサーバの呼び出しを引き起こすわけではありません。例えば、ユーザが [**保存**] をクリックした場合はサーバが呼び出されますが、テキスト・フィールドに入力した場合はサーバの呼び出しは生じません。このため、ユーザがデータ変換を編集していて、[**セッションタイムアウト**] のしきい値より長い間 [**保存**] をクリックしないと、ユーザのセッションは期限切れとなり、保存されていない変更はすべて破棄されます。

自動ログアウトの後、以下のシナリオが生じる可能性があります。

- ログイン・ページが表示されます。
- 管理ポータルがユーザをログアウトさせた直後に再度ログインさせます。これは、Web アプリケーションに [ID でグループ化] の値が設定されており、その結果、自動認証が行われるためです。この場合、管理ポータルの現在のページが更新され、保留中の変更はすべて削除されます。

以下の手順を実行して、ユーザの作業内容が破棄されるのを防ぐことができます。

- ユーザに定期的に作業を保存するよう伝えます。
- ユーザが時間のかかる構成タスク (データ変換の変更など) を行っている Web アプリケーションの [**セッションタイムアウト**] の値を延ばします。[**セッションタイムアウト**] の既定値は 15 分です。

さらに、既定の自動ログアウト動作を保持することをお勧めしますが、ユーザが管理ポータルの [**相互運用性**] のページを表示している場合は、ユーザが自分からログアウトするかブラウザを閉じるまで、ログインしたままにすることができます。これを実行するには、以下のように ^EnsPortal を使用します。

```
^EnsPortal("DisableInactivityTimeout","Portal") = 1
```

注釈　これは、ネームスペースごとの設定です。ログアウト動作を変更するには、ネームスペースごとに個別にこの値を設定する必要があります。

もう一度 ^EnsPortal を使用すると、自動ログアウトを元に戻すことができます。

```
^EnsPortal("DisableInactivityTimeout","Portal") = 0
```

変更を加える前に、生じる可能性のあるセキュリティ上の影響を検討することをお勧めします。

Web アプリケーションとその設定の詳細は、"アプリケーション" を参照してください。

その他のセキュリティ機能

ここでは、その他のセキュリティ機能のいくつかと考慮事項について説明します。以下のとおりです。

- 保護されたデバッグ・シェルの使用の有効化
- メモリ・イメージに存在する機密データの保護

その他のセキュリティ・トピックは、以下を参照してください。

- InterSystems IRIS の構成情報の保護

保護されたデバッグ・シェルの使用の有効化

InterSystems IRIS には、ルーチンを中断し、完全なデバッグ機能をサポートするシェルに入る機能が含まれています ("コマンド行ルーチンのデバッグ" を参照)。InterSystems IRIS には保護されたデバッグ・シェルも含まれています。これには、割り当てられた特権の超越や迂回をユーザができないという利点があります。

既定では、デバッグ・プロンプトのユーザは、現在のレベルの特権を保持します。デバッグ・プロンプトの保護されたシェルを有効にし、このことによってユーザが発行できるコマンドを制限するには、ユーザは **%Secure_Break:Use** 特権 (**%Secure_Break** リソースの **Use** 許可) を保持している必要があります。ユーザにこの特権を付与するには、ユーザを、事前定義の **%SecureBreak** ロールなどの **%Secure_Break:Use** 特権を含むロールのメンバにします。

メモリ・イメージに存在する機密データの保護

エラー状態の中には、“コア・ダンプ” と呼ばれる、プロセス・メモリ・コンテンツのディスク・ファイルへの書き込みを発生させるものもあります。このファイルには、ダンプ時にプロセスで使用されていたすべてのデータのコピーが含まれます (潜在的に重要なアプリケーション・データおよびシステム・データを含む)。これを回避するには、システム全体でコア・ダンプを許可しないようにします。コア・ダンプを許可しないようにする方法は、使用しているオペレーティング・システムによって異なります。詳細は、オペレーティング・システムのドキュメントを参照してください。

インスタンスのセキュリティの強化

インスタンスのセキュリティの強化

InterSystems IRIS® データベースのセキュリティをさらに高めるには、ユーザからのアクセスをより厳密に制限するように構成できます。また、そうすることが必要です。これにより、承認されていないユーザは、ツールを使用したり、機密性の高いリソースにアクセスしたりできなくなります。ここでは、データベース・インスタンスの危険を回避し、セキュリティを向上させるためのさまざまな操作について説明します。ここでは、InterSystems IRIS インスタンスが最小の初期セキュリティでインストールされていることが前提となっています。通常またはロック・ダウンの初期セキュリティを選択した場合、これらの操作の一部は既に自動的に実行されています。

ここでは、インスタンスのセキュリティを強化するための操作の概要について、実行すべき順序で説明します。

- 監査の有効化
- アプリケーションの認証メカニズムの変更
- サービスへのアクセス制限。これには以下が含まれます。
  - 有効なサービス数の制限
  - パブリック・サービス数の制限
  - IP アドレスまたはマシン名を基準にしたサービスへのアクセスの制限
- リモート特権アクセスの制限
- 特権ユーザ数の制限
- _SYSTEM ユーザの無効化
- UnknownUser のアクセスの制限
- サードパーティ・ソフトウェアの構成

インターシステムズのセキュリティ・アドバイザは、インスタンスのセキュリティを強化するために、インスタンスの自動解析機能や推奨される操作を提供します。

重要　InterSystems IRIS データベース・インスタンスには相互依存する要素が多数あります。このため、変更のために指定されていることのみを、過不足なく行うことをお勧めします。例えば、**%All** ロールから (その他の操作を一切行わずに) 単純に UnknownUser を削除すると、最小セキュリティ・インストールで問題が発生します。

監査の有効化

セキュリティの主要な要素は、認証 (Authentication)、承認 (Authorization)、および監査 (Auditing) の“3 つの A”で表されることがよくあります。監査には、以下の 2 つの機能があります。

- セキュリティ・イベントが発生した場合に、何が起こったかに関するデータを提供する。
- 攻撃が追跡、記録され、悪意のある行動に関する証拠がある場合、この機能が存在するとわかっていると、攻撃者に対する抑止力となる。

キー・イベントの監査を有効化する手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、**[システム管理]→[セキュリティ]→[監査]→[監査を有効に]** を選択します。その選択肢が使用できない場合、監査はすでに有効化されています。
2. 管理ポータルのホーム・ページで、**[システムイベントを構成]** ページ (**[システム管理]→[セキュリティ]→[監査]→[システムイベントを構成]**) に移動します。

3. 以下のイベントがまだ有効化されていない場合、**[システムイベントを構成]** ページのそのイベントの行で **[状態変更]** をクリックして、これを有効化します。
   - %System/%DirectMode/DirectMode ー コンソール/ターミナルの使用に関する情報を提供します。コマンド行ユーティリティを重点的に使用するサイトでは、大量のデータが作成される可能性があります。データの増加が問題にならない場合に推奨されます。
   - %System/%Login/Login ー ログインに関する情報を提供します。大規模なサイトでは、大量のデータが作成される可能性があります。データの増加が問題にならない場合に推奨されます。
   - %System/%Login/LoginFailure ー 未承認で試行されたログインに関するフィードバックを提供します。推奨されます。
   - %System/%Security/Protect ー 保護されたデータの読み取り、書き込み、または使用の試行に関するデータを提供します。推奨されます。

アプリケーションの認証メカニズムの変更

データベースへのアクセス制限では、アプリケーションでより厳密な認証メカニズムが使用されるようにインスタンスを構成することが重要です。このセクションでは、この手順の実行方法について説明します。ここでは、サンプル・アプリケーションとして管理ポータルを使用し、より厳密な認証メカニズムへの移行の例として、最小セキュリティ・インストールのような認証なしのアクセスから、パスワードを要求するアクセスへ変更します。

重要　以下の手順を実行すると、変更されるインスタンスが、ポータルへのアクセス以上のさまざまな影響を受ける可能性があります。この詳細は、(1) インスタンスの構成、および (2) この手順のみを実行しているのか、それともここに示しているすべての手順を実行するのかによって異なります。具体的には、以下を行います。

- %Service_WebGateway:Use をパブリックにしないとは、Web アプリケーションのユーザすべてに、何か別の方法で **%Service_WebGateway:Use** を付与する必要があることを意味します。
- UnknownUser を %All ロールから削除すると、さまざまな影響があります。

適切に機能する認証をアプリケーションに提供するには、アプリケーションと、このアプリケーションが使用するサービスの両方に同一の認証メカニズムが必要です。Web アプリケーションでは、Web ゲートウェイ・サービスと一致するように Web ゲートウェイを構成する必要もあります。したがって、管理ポータルに認証を提供するために、同時に機能させる必要のある以下の 3 つのレイヤがあります。

- **%Service_WebGateway** サービス
- Web ゲートウェイ
- 管理ポータル・アプリケーション

これらのレイヤに対応する認証メカニズムがない場合、通常、アクセスが拒否されます。例えば、ログイン・ページの表示や管理ポータルへのアクセスの代わりに、“ページを表示できません” エラーが表示されます。

重要　(1) Web アプリケーションが Web ゲートウェイや **%Service_WebGateway** よりも強力な認証メカニズムを使用している場合に、(2) 認証に成功すると、システムのセキュリティはそれほど強力ではないメカニズムを提供するものになります。

最小セキュリティ・インストールを持つインスタンスでは、Web ゲートウェイ、**%Service_WebGateway**、および管理ポータル・アプリケーションはすべて非認証アクセス用に設定されます。ポータルにパスワード・レベルの認証を提供するには、さまざまな InterSystems IRIS 要素を以下のように構成する必要があります。

- Web ゲートウェイ・サービスは、パスワード認証を要求する必要があります。
- Web ゲートウェイは、この認証のためにユーザ名とパスワードを提供する必要があります。

- このゲートウェイを表すユーザには、Web ゲートウェイ・サービスを使用するために十分な特権が必要です。
- 管理ポータルにはパスワード認証が必要です。
- ポータルのユーザすべてには、このポータルを使用するために十分な特権が必要です。

重要　ポータルでの 1 回のセッション中に、以下の手順をすべて実行します。そうしないと、ポータルからロックアウトされ、残りの手順を ^SECURITY ルーチンで実行しなければならなくなります。

このような変更を行う手順の概要は以下のとおりです。

1. 必要に応じて、インスタンスへの変更を追跡、記録するために監査を有効化します。詳細は、"監査の有効化" を参照してください。
2. CSPSystem ユーザに %Service_WebGateway:Use 特権を付与します。
3. CSPSystem ユーザのパスワードを変更します。
4. 認証用にユーザ名とパスワードを提供するように Web ゲートウェイを構成します。
5. パスワード認証を要求するように %Service_WebGateway を構成します。
6. %Service_WebGateway:Use 特権のパブリック状態を削除します。
7. パスワード認証のみを要求するように、管理ポータル・アプリケーションを構成します。
8. インスタンスのユーザに対して適切な権限レベルを指定します。
9. 必要に応じて、クラス・リファレンスを使用可能にします。
10. 新しいポリシーの施行を開始します。

この処理の完了後、ユーザがポータルに接続しようとすると、ログイン・プロンプトが表示されます。

注意　InterSystems IRIS データベース・インスタンスには相互依存する要素が多数あります。このため、変更のために指定されていることのみを、過不足なく行うことをお勧めします。そうしないと、インスタンスからロックアウトされる場合があります。また、インスタンスが一時的に動作不能になることもあります。

CSPSystem ユーザに %Service_WebGateway:Use 特権を付与

CSPSystem ユーザは、InterSystems IRIS インストール・プロセスにより作成されます。このユーザは、**%Service_WebGateway** サービスとのやり取りにおける Web ゲートウェイを表します。サービスのアクセスは制限されるので、このユーザは、認証プロセスのために **%Service_WebGateway:Use** 特権を持つ必要があります。

注釈　**%Service_WebGateway** と呼ばれるサービスと、**%Service_WebGateway** と呼ばれるリソースがあります。このリソースは、サービスへのアクセスを規制します。したがって、このサービスにアクセスするには、このリソースに対する Use 許可、つまり **%Service_WebGateway:Use** 特権がユーザに必要です。

%Service_WebGateway:Use 特権を CSPSystem ユーザに関連付けるには、以下の手順を実行します。

1. 管理ポータルのホーム・ページで、**[ロール]** ページ (**[システム管理]→[セキュリティ]→[ロール]**) に移動します。
2. **[ロール]** ページで、**[新規ロール作成]** をクリックします。**[ロール編集]** ページが表示されます。このページでは、**[名前]** フィールドが編集可能になっています。
3. **%Service_WebGateway:Use** 特権が含まれるように、ロールの名前を入力します (例 : "GatewayRole")。
4. **[保存]** をクリックします。これにより、InterSystems IRIS にロールが作成されました。
5. **[ロール編集]** ページの **[一般]** タブにある **[特権]** セクションで、**[追加]** をクリックします。このロールについて使用可能なリソースのリストが表示されます。

6. このリストで [%Service_WebGateway] をクリックし、次に [**保存**] をクリックします。新たに作成されたロールには、**%Service_WebGateway:Use** 特権が含まれるようになります。
7. [**ロール編集**] ページの [**メンバ**] タブを選択します。
8. このタブで、新しく作成したロールに CSPSystem ユーザを割り当てることができます。[**使用可能**] リストのユーザから [CSPSystem] をクリックし、右向き矢印をクリックして、[**選択済み**] に移動します。
9. [**割り当てる**] をクリックして、CSPSystem をロールに割り当てます。(つまり、CSPSystem はこのロールのメンバになります)。これは、CSPSystem が **%Service_WebGateway:Use** 特権を持っていることを意味します。

注釈　システムは、Web ゲートウェイを表すために CSPSystem ユーザを作成します。必要であれば、別のユーザがこの機能を実行できます。この手順は CSPSystem ユーザのみを参照します。別のユーザを使用する場合、必要な箇所で CSPSystem をそのユーザ名で置き換えます。

CSPSystem ユーザのパスワードを変更

最小セキュリティ・インストールでは、CSPSystem ユーザに対してパスワード “SYS” が与えられるので、このパスワードを攻撃者の知らないもの、または推測できないものに変更することが重要です。以下はその方法です。

1. 管理ポータルで、[**ユーザ**] ページ ([**システム管理**]→[**セキュリティ**] [**ユーザ**]) に移動します。
2. [**ユーザ**] ページで、[CSPSystem] をクリックします。[**ユーザ編集**] ページが表示されます。
3. [**パスワード**] フィールドに CSPSystem に対する新しいパスワードを入力します。他のユーザは誰もこのパスワードを覚える必要はないので、必要なだけ長く、複雑にすることができます。このパスワードは、次の操作 ″ユーザ名とパスワードを提供するように Web ゲートウェイを構成″ を完了するまでの間、覚えておく必要があります。
4. [**パスワード (再入力)**] フィールドに新しいパスワードを再度入力して、[**保存**] をクリックします。ポータルからエラー・メッセージやダイアログが表示されない場合、パスワードの変更は正常に行われています。

必要に応じて、前の手順で認証のために作成されたロールに CSPSystem が割り当てられていることを確認することもできます。このためには、[**ロール**] タブをクリックします。[**CSPSystem が割り当てられているロール**] という名前の列を持つテーブルには、新しく作成したロールが表示されるはずです。

ユーザ名とパスワードを提供するように Web ゲートウェイを構成

パスワード認証を要求するように **%Service_WebGateway** を構成するため、Web ゲートウェイはユーザ名とパスワードのペアを提供する必要があります。適切な特権レベルを持つユーザをセットアップすると、ゲートウェイが提供できるユーザ名とパスワードのペアが作成されます。次の手順では、InterSystems IRIS サーバから要求されたときに、このユーザ名とパスワードのペアを提供できるようにゲートウェイを構成します。以下はその方法です。

1. 管理ポータルで、[**ウェブゲートウェイ管理**] ページ ([**システム管理**]→[**構成**]→[**ウェブゲートウェイ管理**]) に移動します。
2. [**ウェブゲートウェイ管理**] ページで、左側のリストから [**サーバ接続**] を選択します。[**サーバ・アクセス**] フレームが表示されます。
3. [**サーバ・アクセス**] フレームでは LOCAL サーバがハイライト表示されます。編集のために [**実行**] をクリックします。[サーバ・アクセス] パラメータや [エラー・ページ] パラメータの並んだページが表示されます。
4. このページには、[**接続セキュリティ**] セクションがあります。
5. [**接続セキュリティ・レベル**] ドロップダウンに [パスワード] が表示されていることを確認します。
6. [**ユーザ名**] フィールドに、「CSPSystem」と入力します。
7. [**パスワード**] および [**パスワード (確認)**] フィールドに、前のセクションで選択したパスワードを入力します。
8. ページの下部にある [**設定を保存**] をクリックします。

9. 管理ポータルに戻るには、左ペインにあるリスト下部の [**管理ポータルに戻る**] をクリックします。

パスワード認証を要求するように %Service_WebGateway を構成

ユーザ名とパスワードを提供するようにゲートウェイを構成し、CSPSystem ユーザに必要なレベルの特権を付与したら、次に、パスワード認証を要求するように、Web アプリケーションを管理するサービス (**%Service_WebGateway**) を構成します。以下はその方法です。

1. 管理ポータルのホーム・ページで、[**サービス**] ページ ([**システム管理**]→[**セキュリティ**]→[**サービス**]) に移動します。
2. [**サービス**] ページで、[%Service_WebGateway] をクリックします。**%Service_WebGateway** の [**サービス編集**] ページが表示されます。
3. [**サービス編集**] ページの [**許可された認証方法**] で [**認証なし**] アクセスが無効化されていること、および [**パスワード**] アクセスが有効化されていること (別名 : “ログイン認証”) を確認します。[**保存**] をクリックします。

%Service_WebGateway:Use 特権のパブリック状態を削除

%Service_WebGateway がパスワード認証を要求し、適切な権限を持つユーザを使用してゲートウェイで認証を行うことができるようになったら、次に、**%Service_WebGateway:Use** をパブリック許可から除外します。以下はその方法です。

1. 管理ポータルのホーム・ページで、[**リソース**] ページ ([**システム管理**]→[**セキュリティ**]→[**リソース**]) に移動します。
2. [**リソース**] ページで、**%Service_WebGateway** に対応する行の [**編集**] をクリックします。**%Service_WebGateway** の [**リソース編集**] ページが表示されます。
3. [**パブリック許可**] セクションで [**使用**] ボックスをオフにします。[**保存**] をクリックします。

重要　**%Service_WebGateway:Use** がパブリック特権ではなくなると、これが明示的に付与されたユーザのみが Web アプリケーションを使用できるようになります。これらのユーザのリストを作成し、その他の方法でこの特権を付与する必要が生じる場合もあります。

パスワード認証のみを受け入れるように管理ポータルを構成

ゲートウェイと InterSystems IRIS サーバの間の接続に新しい認証メカニズムが使用されるようになったら、次に、これに見合ったメカニズムが使用されるように管理ポータル・アプリケーションを構成します。この例ではインスタンス認証メカニズムを使用します。ポータルの認証メカニズムを変更するための手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、[**ウェブ・アプリケーション**] ページ ([**システム管理**]→[**セキュリティ**]→[**アプリケーション**]→[**ウェブ・アプリケーション**]) に移動します。
2. [**ウェブ・アプリケーション**] ページでは、/csp/sys アプリケーションは管理ポータルのホーム・ページを表します。このアプリケーションを編集するには、この行の名前 [**/csp/sys**] をクリックします。/csp/sys アプリケーションの [**ウェブ・アプリケーションの編集**] ページが表示されます。
3. [**セキュリティの設定**] の [**許可された認証方法**] で、[**認証なし**] アクセスを無効化し、[**パスワード**] アクセスを有効化します。[**保存**] をクリックします。
4. ポータルのその他のページと選択肢を構成するすべてのアプリケーションに対しても、[**非認証**] アクセスを無効にし、[**パスワード**] アクセスを有効にします。これらのアプリケーションには以下のものがあります。
   - /csp/sys/exp
   - /csp/sys/mgr
   - /csp/sys/op
   - /csp/sys/sec

注釈　アプリケーション /csp/sys/op を編集した後、さらなる変更を加えるためには認証が必要になります。

このように構成することで、ポータルを使用するにはパスワード認証 (別名は “インスタンス認証”) が必要になり、認証なしアクセスが許可されなくなるので、各構成部分が整合性のある動作をします。次の手順では、関連するユーザがすべて、ポータルへの適切なアクセス権を持つことを確認します。

インスタンスのユーザに対して適切な権限レベルを指定

認証なしアクセスを受け入れるようにポータルが構成されている場合、どのようなユーザでも UnknownUser として接続できます。最小セキュリティ・インストールでは UnknownUser は **%All** ロールのメンバになるので、ポータルからロックアウトされる心配はありません。ここで、ポータルによりパスワード認証が要求されるようになると、正当なユーザは **%Operator** ロール、**%Manager** ロール、または **%All** ロールのメンバとなる必要があります。

最小セキュリティ・インストールでは、SuperUser、Admin、_SYSTEM、および UnknownUser はすべて、このレベルの特権を持ちます。また、パスワードはすべて “SYS” です。

注釈　通常のインストールまたはロック・ダウン・インストールでは、UnknownUser は有効になりますが、ロールは割り当てられません。

通常のインストールまたはロック・ダウン・インストールでは、パスワードはインストール・プロセスで設定されますが、ここで変更することもできます。

適切にユーザのセキュリティを確保する手順は以下のとおりです。

1. UnknownUser を無効化します。または、**%All** ロールから UnknownUser を削除します。
   - UnknownUser を無効化する手順は以下のとおりです。
     a. [ユーザ] ページ ([**システム管理**]→[**セキュリティ**]→[**ユーザ**]) で、[**名前**] 列の下の [UnknownUser] をクリックします。UnknownUser に対応する [**ユーザ編集**] ページが表示されます。
     b. [**ユーザ有効**] フィールドをクリアし、[**保存**] をクリックします。
   - **%All** ロールから UnknownUser を削除するには、以下の手順を実行します。
     a. [**ユーザ**] ページ ([**システム管理**]→[**セキュリティ**]→[**ユーザ**]) で、[**名前**] 列の下の [UnknownUser] をクリックします。UnknownUser に対応する [**ユーザ編集**] ページが表示されます。
     b. [**ユーザ編集**] ページの [**ロール**] タブに進みます。
     c. [**ユーザ UnknownUser には以下のロールが割り当てられています**] テーブルの **[%All]** 行で、[**削除**] をクリックします。

   重要　UnknownUser を通じたアクセスの制限は広い範囲に影響を及ぼします。これは、インスタンスのユーザが十分な特権を持っていない場合に特に顕著です。

2. 認証されていない可能性のあるその他のユーザが、**%All**、**%Developer**、**%Manager**、**%Operator**、**%SQL**、または特権を付与されたその他のユーザ定義ロールのメンバではないことを確認します。このためには、**%All** ロールから UnknownUser を削除するのと類似した処理を行います

   (特権を付与するユーザ定義ロールは、**%Admin...** リソースのいずれか、**%Development**、または **%Service** もしくは **%System** リソースのいずれかに対する Use 許可を持つか、**%DB_IRISLIB** もしくは **%DB_IRISSYS** に対する Write 許可を持つ可能性があります)。

3. ポータルへのアクセス権を持っているはずのユーザがすべて、**%All**、**%Developer**、**%Manager**、**%Operator**、**%SQL**、またはポータルへのアクセスを付与する任意のユーザ定義ロールに割り当てられていることを確認します。これらのユーザそれぞれについて、以下の手順を実行します。

a. **[ユーザ]** ページ (**[システム管理]→[セキュリティ]→[ユーザ]**) で、**[名前]** 列の下にあるユーザの名前をクリックします。そのユーザに対応する **[ユーザ編集]** ページが表示されます。

b. **[ユーザ編集]** ページの **[ロール]** タブに進みます。

c. 目的のロールを **[使用可能]** から **[選択済み]** リストに移動します。このためには、ロールを選択し、右矢印ボタンをクリックし、**[割り当てる]** をクリックして、ユーザをロールに割り当てます。

4. SuperUser および Admin ユーザのパスワードを既定値から変更し、これらのアカウントを無効にします。以下はその方法です。

a. **[ユーザ]** ページ (**[システム管理]→[セキュリティ]→[ユーザ]**) で、**[名前]** 列の下にあるユーザの名前をクリックします。そのユーザに対応する **[ユーザ編集]** ページが表示されます。

b. **[新規パスワード入力]** をクリックします。

c. **[パスワード]** フィールドに新しいパスワードを入力します。

d. **[パスワード (再入力)]** フィールドでパスワードを再入力します。

e. **[ユーザ有効]** の選択をクリアし、**[保存]** をクリックします。

注釈 InterSystems IRIS には、**%All** ロールを持つ有効なアカウントが 1 つ以上必要です。**%All** ロールを持つ一意のユーザを作成し、SuperUser、Admin、および _SYSTEM ユーザは無効にすることをお勧めします。

重要 少なくとも 1 人のポータル管理ユーザのパスワードを知っていることを確認してください。そうしないと、ポータルからロックアウトされ、緊急アクセスを使用してログインし、^SECURITY ルーチンを使って 1 つ以上のパスワードをリセットしなければならなくなる可能性があります。

クラス・ドキュメントを使用可能にする

サービス、Web ゲートウェイ、およびポータル・アプリケーションの構成が完了したら、クラス・ドキュメントを使用できるようにしましょう。以下はその方法です。

1. 管理ポータルのホーム・ページで、**[ウェブ・アプリケーション]** ページ (**[システム管理]→[セキュリティ]→[アプリケーション]→[ウェブ・アプリケーション]**) に移動します。

2. ドキュメントを使用できるようにするには、以下の操作を実行します。

a. **[ウェブ・アプリケーション]** ページでは、/csp/documatic アプリケーションがクラスリファレンス・アプリケーションです。このアプリケーションを編集するには、この行の **[/csp/documatic]** をクリックします。/csp/documatic アプリケーションの **[ウェブ・アプリケーションの編集]** ページが表示されます。

b. **[セキュリティの設定]** の **[許可された認証方法]** で、**[認証なし]** アクセスを無効化し、**[パスワード]** アクセスを有効化します。**[保存]** をクリックします。

注釈 通常のインストールの場合、パスワード・アクセスは既に有効になっています。

この手順を実行していない場合、サービスはパスワード・プロンプトを要求しますが、アプリケーションは認証なしアクセスを試行します。これにより、%All に割り当てられているユーザを含め、すべてのユーザがドキュメントにアクセスできなくなります。

新しいポリシーの施行を開始

この時点で、InterSystems IRIS インスタンスの構成は完了し、適切な動作をするようになっています。ただし、既存のすべての接続で、引き続き非認証アクセスが使用されています。新しいポリシーの施行を開始するには、次のイベントが発生しなければなりません。

- Web ゲートウェイは、認証接続を確立する必要があります。
- また、すべてのユーザも認証接続を確立する必要があります。

認証された Web ゲートウェイ接続の確立

Web ゲートウェイに認証接続を確立させるには、次の手順を実行します。

1. 管理ポータルのホーム・ページで、**[システム管理]→[構成]→[ウェブゲートウェイ管理]** を選択します。**[ウェブゲートウェイ管理]** ページが表示されます。
2. **[ウェブゲートウェイ管理]** ページで、左側のリストから **[接続を閉じる]** を選択します。**[接続を閉じる]** フレームが表示されます。
3. **[接続を閉じる]** をクリックします。これにより、ゲートウェイと InterSystems IRIS サーバ間の接続がすべて閉じられたことを示すメッセージが表示されます。

次回ユーザがページを要求すると、ゲートウェイにより、InterSystems IRIS サーバへの接続が再度確立されます。この接続では、選択された認証メカニズムが使用されます。

認証されたユーザ接続の確立

この時点では、管理ポータルへのすべての接続で、依然として認証なしアクセスが使用されています。すぐに認証アクセスが必要でない場合、何もする必要はありません。ユーザはポータルへの接続を順次終了し、再接続するときに認証が必要となります(接続の終了する理由には、マシンの再起動、ブラウザの停止と再開、ブラウザ・キャッシュのクリア、ポータルのログアウトなどがあります)。

接続で強制的に認証アクセスを使用する必要がある場合、以下のようにして InterSystems IRIS を停止し、再起動します。例えば、Windows で、InterSystems IRIS が既定の [スタート] メニューのページで利用できる場合、以下のようにします。

1. Windows の [スタート] メニューから **[プログラム]→[InterSystems IRIS]** を選択し、InterSystems IRIS インスタンスを再起動します。
2. InterSystems IRIS インスタンスのサブメニューで、**[インターシステムズの停止]** を選択します。
3. 表示されたダイアログで、**[再起動]** を選択し、[OK] をクリックします。

注釈　InterSystems IRIS の実稼動インスタンスを使用している場合、ユーザは一時的に、InterSystems IRIS 全体またはポータルへアクセスできなくなるため、再起動にはトラフィックが少ない時間を選択します。

パブリック・リソース数の制限

どのようなリソースでも、パブリック・リソースとして指定できます。つまり、どのようなユーザでも、パブリック設定に応じて、このリソースを読み取り、書き込み、または使用できます。以下のリソースおよび許可は、常にパブリックにする必要があります。

テーブル A-2: 必須のパブリック・リソースおよびその許可

リソース	許可
%DB_IRISLOCALDATA	R
%DB_IRISLIB	R
%DB_IRISTEMP	RW

インスタンスのセキュリティを強化する場合は、パブリック・リソースの数を制限してください。そのための手順は以下のとおりです。

1. これらのリソースへのアクセスを本当に必要としているすべてのユーザに、必要な特権が与えられていることを確認します。

   重要　**%Service_WebGateway:Use** に対する特権を適切なユーザに提供しなかった場合、この手順により、管理ポータルやその他の Web アプリケーションからの大規模なロックアウトが発生する場合があります。

2. 管理ポータルのホーム・ページで、[**リソース**] ページ ([**システム管理**]→[**セキュリティ**]→[**リソース**]) に移動します。
3. リソースに対するパブリック許可が 1 つ以上ある場合、各リソースの所有する権限は、[**リソース**] ページのリソース・テーブルにある [**パブリック許可**] 列にリストされます。[**編集**] をクリックしてリソースを選択します。選択したリソースの [**リソースの編集**] ページが表示されます。
4. [**リソースの編集**] ページで、チェックマークが付けられている [**パブリック許可**] フィールドをすべてクリアし、[**保存**] をクリックします。このリソースはパブリックではなくなります。

すべてのパブリック・リソースについて、この操作を実行します。

サービスへのアクセス制限

ユーザが InterSystems IRIS と対話する経路にはさまざまな種類があります。サービスは、これらの経路へのアクセスを規制します。インターシステムズのサービスへのアクセスを制限するには、以下の選択肢があります。

- 有効なサービス数の制限。使用しているアプリケーションで必要なもののみにします。
- パブリック・サービス数の制限。使用しているアプリケーションで必要なもののみにします。
- IP アドレスまたはマシン名を基準にしたサービスへのアクセスの制限

有効なサービス数の制限

有効なサービス数を制限するには、以下の手順を実行します。

1. InterSystems IRIS インスタンスで必要なサービスを判断します。通常、これらは以下のとおりです。
   - ユーザ・アクセスの各形式で必要とされるサービス
   - 自動アクセスで必要とされるサービス
   - ローカルなプログラマ・モード・アクセスのための **%Service_Console** (Windows の場合) または **%Service_Terminal** (UNIX または UNIX® の場合)
2. 管理ポータルのホーム・ページで、[**サービス**] ページ ([**システム管理**]→[**セキュリティ**]→[**サービス**]) に移動します。
3. [**サービス**] ページで、必要のないサービスそれぞれの名前をクリックして選択します。選択したサービスの [**サービス編集**] ページが表示されます。
4. [**サービス編集**] ページで、[**サービス有効**] フィールドをクリアし、[**保存**] をクリックします。これで、このサービスは無効化されました。

必要のないサービスをすべて無効化すると、InterSystems IRIS への経路はサービスに必要な経路のみになります。

パブリック・サービス数の制限

各サービスはリソースに対応します。ほとんどの場合、リソースとサービスは同じ名前を持ちます (例 : **%Service_WebGateway**)。ただし、**%Service_Bindings** サービスは例外で、**%Service_Object** リソースおよび **%Service_SQL** リソースと関連付けられています。サービスは、それに関連付けられているリソースの設定のため、パブリックです。したがって、サービスを非パブリックにする手順は、その他のリソースを非パブリックにする手順と同じです。これについては、"パブリック・リソース数の制限" で説明しています。

IP アドレスまたはマシン名を基準にしたサービスへのアクセスの制限

一部のサービスでは、IP アドレスやマシン名に従って、サービスへのアクセスを制限することができます。これは、“許可された接続”を制限する機能と言われます。この機能をサポートしているサービスは以下のとおりです。

- **%Service_Bindings**
- **%Service_CacheDirect**
- **%Service_ECP**
- **%Service_Monitor**
- **%Service_Shadow**
- **%Service_WebGateway**

既定では、サービスはすべてのマシンからの接続を受け入れます。サービスにアドレスやマシン名が関連付けられていない場合、このサービスはすべてのマシンからの接続を受け入れます。サービスが接続を受け入れるアドレスやマシン名が 1 つ以上指定されている場合、サービスはこれらのマシンからの接続のみ受け入れます。

この機能は、**%Service_CallIn**、**%Service_ComPort**、**%Service_Console**、**%Service_DataCheck**、**%Service_Login**、**%Service_Mirror**、**%Service_Telnet**、および **%Service_Terminal** では使用できません。

IP アドレスを基準にサービスへのアクセスを制限するには、以下の手順に従います。

1. サービスへの正当なアクセスを持つマシンの IP アドレスを判断します。
2. 管理ポータルのホーム・ページで、[**サービス**] ページ ([**システム管理**]→[**セキュリティ**]→[**サービス**]) に移動します。
3. [**サービス**] ページで、IP アドレスを基準にアクセスを制限するサービスの名前を個別にクリックして選択します。選択したサービスの [**サービス編集**] ページが表示されます。
4. [**サービス編集**] ページの [許可された接続] セクションで、[**新規追加**] をクリックします。
5. 表示されたダイアログに、接続を許可する IP アドレスを入力します。[OK] をクリックします。
6. [**新しく追加**] をクリックし、必要なアドレスを入力します。

接続を許可する IP アドレスを制限するサービスそれぞれについて、この手順を実行します。

リモート特権アクセスの制限

InterSystems IRIS は、ECP リモート・ジョブ要求をサポートしています。ただし、リモート・ジョブはサーバ上で root として実行されるので、意図した以上の特権でユーザがサーバ上で作業できる可能性があります。リモート・ジョブの扱いを無効にして、サーバへのリモート特権アクセスを制限するには、″このパラメータの変更″ の手順に従い、**netjob** を `false` に設定します。既定ではこの設定は `true` になっています。

特権ユーザ数の制限

すべての InterSystems IRIS インスタンスには、**%All** ロールに割り当てられたユーザが少なくとも 1 人必要です。実際、このロールに割り当てられたユーザが 1 人のみの場合、InterSystems IRIS はこのロールからこのユーザを削除できないようにします。しかし、時間の経過に従って、あるインスタンスの **%All** に必要以上のユーザが割り当てられてしまうことがあります。その原因には、割り当てられたユーザは組織を離れたがアカウントが無効化されていない、一時的な割り当てが削除されていないなどがあります。

%All ロールと共に、システム定義ロールの **%Manager**、**%Developer**、**%Operator**、および **%SQL** もユーザに過度の特権を与えることができます。また、このような動作を行うユーザ定義ロールもあります。このようなロールに割り当てられたユーザは、“特権ユーザ”と呼ばれることもあります。

特権ユーザの数を制限するには、どのユーザが各特権ロールに割り当てられているかを判断し、不要なユーザを削除します。以下はその方法です。

1. 管理ポータルのホーム・ページで、[**ロール**] ページ ([**システム管理**]→[**セキュリティ**]→[**ロール**]) に移動します。
2. [**ロール**] ページで、ロールの名前をクリックします。そのロールに対応する [**ロール編集**] ページが表示されます。
3. [**ロール編集**] ページの [**メンバ**] タブをクリックします。そのロールに割り当てられているユーザとロールのリストが表示されます。
4. 指定されたロールからユーザを削除するには、削除するユーザまたはロールの行にある [**削除**] をクリックします。

%All および前述のその他のロールを含め、特権ロールそれぞれについてこの手順を実行します。また、_SYSTEM ユーザを無効化することも重要です。その手順については、″_SYSTEM ユーザの無効化″ で説明します。

重要　一見して特権のないロールが “代理特権” とも呼べる特権を持っていることもあります。この現象は、一見して特権のないロールを特権ロールに割り当てているときに発生します。この場合、代理特権を持つロールに割り当てられたすべてのユーザは、特権ロールに関連付けられたすべての特権を持ちます。

可能な限り、代理特権は作成しないようにしてください。どうしても避けられない場合、代理特権を持つロールに割り当てるユーザの数はできる限り少なくします。

_SYSTEM ユーザの無効化

InterSystems IRIS インストール・プログラムは _SYSTEM ユーザを作成します。このユーザは、SQL 標準に従って、SQL ルート・ユーザとして作成されます。最小セキュリティ・インストールでは、このユーザの既定のパスワードは “SYS” です。標準およびロックダウン・インストールの既定のパスワードは、インストール処理中に指定されたものになります。このユーザとパスワード “SYS” がどちらも SQL 標準により公開されているため、またこのユーザの SQL 特権のため、_SYSTEM を無効化することは、InterSystems IRIS インスタンスへのアクセスを制限するために重要です。

そのための手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、[**ユーザ**] ページ ([**システム管理**]→[**セキュリティ**]→[**ユーザ**]) に移動します。
2. [**ユーザ**] ページで、名前 [**_SYSTEM**] をクリックして、_SYSTEM の [**ユーザ編集**] ページを開きます。
3. _SYSTEM の [**ユーザ編集**] ページで、[**ユーザ有効**] フィールドをクリアします。[**保存**] をクリックします。

注釈　_SYSTEM を無効化した後でルート・レベルの SQL 特権を確認する必要がある場合は、必要な操作を実行できるように、ユーザを一時的に有効化する必要があります。

UnknownUser のアクセスの制限

認証なしアクセスをサポートしているインスタンスでは、認証を使用しない接続は UnknownUser アカウントを使って確立されます。最小セキュリティ・インストールでは、既定の動作は以下のようになります。

- すべての接続で UnknownUser が使用されます。
- UnknownUser は **%All** ロールに割り当てられます。
- UnknownUser は SQL 特権をすべて保持しています。

UnknownUser のアクセスを制限するには、有効なすべてのサービスの認証なしアクセスを無効にします(その他の操作は効果がないか、管理ポータルからロックアウトされる可能性があります)。

注釈　このセクションでこれまでに示したすべての操作の実行を完了している場合、既に UnknownUser の無効化とパブリック・リソース数の制限が完了している可能性があります。

UnknownUser アカウントで発生する可能性のあるロックアウトの問題

あるインスタンスが最小セキュリティでインストールされている場合、UnknownUser のロールは **%All** になります。また、このインスタンスは、すべてのサービスおよびアプリケーションに対して、認証なしアクセスを提供します。このユーザのロールを単に **%All** から別のものに変更しても、認証なしアクセスを引き続き許可している場合は、基本機能を使用できない可能性があります。

これは、このような状況では、認証が行われないまま、InterSystems IRIS により、選択したツールへの接続が確立されるからです。認証が行われないと、システムにより、自動的にユーザ・アカウントが UnknownUser に設定されます。次に、ユーザ特権がチェックされます。UnknownUser が十分な特権を持っていない場合、ツールへのアクセスは制限されるか不可能になります。このような状況では、例えば、ターミナルには“アクセスが拒否されました”というメッセージが表示され、シャットダウンされます。ポータルではメイン・ページは表示されますが、オプションは一切選択できません。

この状態を正常に戻すには、以下の手順に従います。

1. InterSystems IRIS を緊急アクセス・モードで起動します。
2. UnknownUser アカウントに十分な特権を与えます。

UnknownUser を使用できないようにするには、管理ポータルに対する認証メカニズムをアップグレードする必要があります。

サードパーティ・ソフトウェアの構成

インターシステムズ製品は、ウイルス・スキャンなどのインターシステムズ製ではないツールと共に実行したり、そのようなツールとやり取りすることが頻繁にあります。このようなやり取りでもたらされる可能性がある影響に関する重要な情報については″インターシステムズ製品と連係して動作するようにサードパーティ・ソフトウェアを構成する方法″を参照してください。

セキュリティ・アドバイザ

セキュリティ・アドバイザ

InterSystems IRIS システムの保護においてシステム・マネージャを支援するために、InterSystems IRIS 管理ポータルにはセキュリティ・アドバイザと呼ばれるツールが組み込まれています。これは、セキュリティに関連してシステム構成に収められている現在の情報を表示する Web ページです。セキュリティ・アドバイザでは、推奨される変更点や見直すべき領域が示され、推奨される変更を行うための管理ポータル内のページへのリンクが提供されます。

重要　セキュリティ・アドバイザが提供するのは一般的な推奨内容であり、そこではインスタンス固有のニーズや要件は考慮されていません。InterSystems IRIS のインスタンスにはそれぞれ固有の要件と制約がある点を念頭に置くことは重要です。セキュリティ・アドバイザには、目的のインスタンスに無関係な問題が表示されることもあれば、重要度の高い問題が表示されないこともあります。例えば、サービスで Kerberos 認証のみを使用することがセキュリティ・アドバイザで推奨されていても、実際の稼動環境によっては、オペレーティング・システムによる認証やインスタンス認証、さらには非認証のアクセスが適切な場合もあり得ます。

セキュリティ・アドバイザには、以下のような一般的な機能があります。

- **[詳細]** ボタン ー 各選択項目には **[詳細]** ボタンがあります。このボタンを選択すると、その選択項目に関連する InterSystems IRIS の詳細を管理するためのページが、セクションの制限内容に応じて表示されます。
- **[名前]** ボタン ー 各セクションで指定されている項目は、それぞれがリンクとして表示されます。これらの項目のいずれかを選択することで、その項目を管理するためのページが表示されます。
- **[無視]** チェック・ボックス ー 各セクションで指定されている項目ごとに、その項目に関連付けられた **[無視]** チェック・ボックスがあります。項目が特定の要件に該当しないと判断した場合にこのチェック・ボックスにチェックを付けると、指定した項目の行がグレー表示になります。セキュリティ・アドバイザの推奨に従って InterSystems IRIS を設定している場合は、**[無視]** チェック・ボックスの設定に関係なく、この行は表示されなくなります。

監査

このセクションには、監査そのものおよび特定の監査イベントに関する推奨事項が表示されます。これには以下のものがあります。

- 監査を有効にするべきです ー 監査を実行すると、注意の必要なシステム・イベントや異常なシステム・イベントが発生した後で、検討作業に有用な情報を収めた記録が作成されます。
- この種類の監査イベントは有効にするべきです ー 特定のイベントを監査することで、さまざまなトピックに関する詳細な情報が得られます。特に、監査が有効になっていないときに注目されるイベントは以下のとおりです。
  - DirectMode イベント ー このイベントを監査することで、ユーザに重大な特権を与える InterSystems IRIS 接続に関する情報が得られます。
  - Login イベント ー このイベントを監査することで、疑義のあるログインに関する情報が得られます。
  - LoginFailure イベント ー このイベントを監査することで、システムに対する不適切なアクセス権を得ようとする操作に関する情報が得られます。

サービス

ここでは、インターシステムズのサービスに関する推奨事項について説明します。セキュリティ・アドバイザでは、サービスごとにその設定に応じて以下の点が指摘されます。

- %グローバルを更新できる設定は無効にするべきです ー パーセントで始まるグローバルにはシステム情報が保持されていることが多いので、ユーザがこれらのグローバルを操作できるようになっていると、深刻で広範囲に及ぶ予測不能な影響が出る可能性があります。

- 非認証は無効にされるべきです ー 未認証の接続があると、身元が不確かな UnknownUser アカウントを含むすべてのユーザが、該当のサービスを通じて InterSystems IRIS に無制限にアクセスできます。
- 要求があるまでサービスを無効にするべきです ー セキュリティ・アドバイザで監視されているサービスを介したアクセスでは、システムに対する過剰なレベルのアクセスが可能になります。
- サービスは Kerberos 認証を使用するべきです ー Kerberos 以外の認証メカニズムを通じたアクセスでは、Kerberos 以上のレベルのセキュリティ保護が得られません。
- サービスにはクライアント IP アドレスを割り当てるべきです ー 接続を受け入れる IP アドレスの数を制限することで、より確実に InterSystems IRIS への接続を監視できるようになります。
- サービスはパブリック ー パブリック・サービスがあると、身元が不確かな **UnknownUser** アカウントを含むすべてのユーザが、そのサービスを通じて InterSystems IRIS に無制限にアクセスできます。

ロール

このセクションでは、過剰な特権を持っている可能性のあるすべてのロールに関する推奨事項について説明します。それ以外のロールについては取り上げません。ロールごとに、セキュリティ・アドバイザでは以下の点が指摘されます。

- ロールが監査データベースに対する権限を保持しています ー 監査データベースに対する読み取りアクセスによって、不適切な範囲まで監査データが公開される可能性があります。また、書き込みアクセスによって、監査データベースにデータが不適切に挿入される可能性があります。
- このロールは **%Admin_Secure** 権限を所有しています ー この権限を使用すると、アセットに対するユーザのアクセスを設定、変更、および拒否できます。また、セキュリティ関連の他の機能を変更できます。
- このロールは %IRISSYS データベースの書き込み権限を所有しています ー %IRISSYS データベースに対する書き込みアクセスによって、システムのコードおよびデータが漏洩する可能性があります。

ユーザ

ここでは、ユーザ全般に関する推奨事項および個々のユーザ・アカウントに対する推奨事項について説明します。この領域では、セキュリティ・アドバイザで以下の点が指摘されます。

- 少なくとも 2 名から最大 5 名のユーザが **%All** ロールを保持する必要があります ー **%All** ロールを持つユーザが少なすぎると、緊急時にアクセス上の問題につながることがあります。また、多すぎると、システムの公開性が高くなりすぎて機密漏洩につながる可能性があります。
- このユーザは **%All** ロールを所有しています ー どのユーザが %All ロールを持っているかを明示的に公表することで、無関係なユーザが %All ロールを持つことを防止できます。
- UnknownUser アカウントは **%All** ロールを持つべきではありません ー 匿名のユーザがすべての権限を持っていると、システムのセキュリティが確保できません。最小のセキュリティ・レベルを持つインスタンスでは、UnknownUser アカウントに %All ロールが与えられていますが、このようなインスタンスのセキュリティを計画的に確保することはできません。
- アカウントが使用されていません ー 承認されないアクセスを得ようとする侵入者にとって、使用されていないアカウントは絶好の侵入ポイントになります。
- アカウントが休眠状態のようですので無効にするべきです ー 承認されないアクセスを得ようとする侵入者にとって、休止状態のアカウント (31 日以上使用されていないアカウント) は絶好の侵入ポイントになります。
- パスワードを既定のパスワードから変更するべきです ー 既定のままのパスワードは、承認されないアクセスを得ようとする侵入者によって侵入ポイントとしてよく利用されます。

Web アプリケーション、特権ルーチン・アプリケーション、およびクライアント・アプリケーション

アプリケーションごとに専用のセクションがあり、そこではそれぞれのアプリケーション・タイプの詳細を容易に確認できます。これらのセクションには、アプリケーションへのアクセスおよびアプリケーションによって与えられている特権に関連する推奨事項が表示されます。この領域では、セキュリティ・アドバイザで以下の点が指摘されます。

- アプリケーションがパブリックです ― パブリックなアプリケーションがあると、身元が不確かな **UnknownUser** アカウントを含むすべてのユーザが、そのアプリケーションに関連付けられたデータおよびそのアプリケーションでサポートされているアクションに無制限にアクセスできます。アプリケーションによって **%All** ロールも与えられている場合は、それが条件付きでも無条件でも、この影響はさらに大きくなります。
- 条件によりアプリケーションは **%All** ロールを付与します ― ユーザがすべての権限を持つ可能性があると、システムのセキュリティを確保できなくなります。アプリケーションがパブリックでもある場合、この影響はさらに大きくなります。
- アプリケーションは **%All** ロールを付与します ― ユーザがすべての権限を持っていると、システムのセキュリティを確保できなくなります。アプリケーションがパブリックでもある場合、この影響はさらに大きくなります。

インターシステムズのプロセスおよびオペレーティング・システム・リソースの保護

概要

このドキュメントでは、InterSystems IRIS® データ・プラットフォームのインスタンスを実行しているオペレーティング・システムのセキュリティを強化することによって、侵入者の攻撃対象となり得る領域を減らす方法を説明します。トピックは以下のとおりです。

- InterSystems IRIS インスタンスで必要なオペレーティング・システム・サービス
- さまざまなタイプの InterSystems IRIS プロセス、および各プロセスの目的
- 実行中インスタンス内の InterSystems IRIS プロセスの機能を特定する方法
- 自身のサイトには不要と思われるオプションの InterSystems IRIS プロセスを削除または無効化する方法
- UNIX® 上の **iris** プロセスまたは Windows 上の **irisdb.exe** プロセスに加えて、実行中のインスタンスに必要なプロセス
- InterSystems IRIS プロセスに使用する TCP ポートと UDP ポート、および各ポートの目的

InterSystems IRIS プロセス

InterSystems IRIS インスタンスが含まれているほとんどのプロセスでは、UNIX® の場合は iris 実行可能ファイル、Windows の場合は irisdb.exe 実行可能ファイルが使用されます。これらの実行可能ファイルはそれぞれ、インストール・ディレクトリ下の **bin** ディレクトリにあります。実行中のインスタンスは、さまざまなシステム・プロセスを使用して、ユーザ・コードを実行しているプロセスを調整およびサポートします。InterSystems IRIS プロセスは、管理ポータルで **[システムオペレーション]→[プロセス]** に移動して調べることができます。

コア・プロセス

コア・システム・プロセスは、インスタンス初期化の早期段階で開始され、**User** 列に値を持っていません。これらのプロセスは **Routine** 列の値によって識別できます。システム・プロセスの場合、この列に InterSystems IRIS ルーチンの名前が常に含まれているわけではありません。**Routine** 列には、以下のコア・システム・プロセスが名前別に表示されます。

- CONTROL – 共有メモリを作成および初期化して、各種の制御関数を提供します。
- WRTDMN – データベースおよび WIJ へのすべての書き込みを実行します (ライト・デーモン)。
- GARCOL – サイズの大きいキルをガーベッジ・コレクションします。
- JRNDMN – ジャーナル書き込みを実行します。
- EXPDMN – データベース拡張を実行します。
- AUXWD – ライト・デーモン・タスクを実行します (ライト・デーモン予備ワーカ)。
- MONITOR – アラートをアラート・ファイルに書き込んで、電子メール・アラートを送信します。
- CLNDMN – 停止しているプロセスを検知して、立ち往生しているリソースをクリーンアップします。
- RECEIVE – ECP ワーカ・プロセスを管理します。
- ECPWork – ECP タスクを実行します (ECP ワーカ・プロセス)。
- %SYS.SERVER – TCP 要求を受け取り、それらの要求を処理するようにワーカをディスパッチします (スーパーサーバ・プロセス)。
- %CSP.Daemon – Web セッションの期限切れを管理します。

- LMFMON – InterSystems IRIS ライセンスを監視して、使用状況データを UDP 経由でライセンス・サーバに送信します。
- %SYS.Monitor.xxx – システム監視タスクを実行します (さまざまなシステム監視ワーカ)。
- SYS.Monitor.xxx – アラートをアラート・ファイルに書き込んで、電子メール・アラートを送信します。

コア・システム・プロセスを停止することはできません。これらのプロセスを停止すると、InterSystems IRIS インスタンスは正常に動作できなくなります。

他のさまざまな InterSystems IRIS システム・プロセスがコア・システム・プロセスの後に開始されます。これらの多くは動的に開始されます。これらのプロセスについては、User 列に値が表示されます。これらのプロセスの多くは必須ではないため、必要な場合や構成されている場合を除いて開始されません。これらのプロセスは通常、プロセス表示の Routine、User、および Client EXE の各列の値によって識別できます。

インスタンス開始時に、タスク・マネージャ・プロセス (TASKMGR) が作成されます。このプロセスは、各種のスケジュールされたシステム定義タスクとユーザ定義タスクを開始して、次の設定に基づいて実行されます。

- ユーザ名 = TASKMGR
- ルーチン = %SYS.TaskSuper.1
- オペレーティング・システム・ユーザ名 = TASKMGR

ECP を使用していない場合、次の手順を実行することで、ECPWork プロセスが開始されることを防止できます。

1. 管理ポータルから、**[システム管理]→[構成]→[接続性]→[ECP設定]** を選択して、アプリケーション・サーバとデータ・サーバの最大数をゼロに設定します。
2. ECP サービスを無効にします。

ECP サーバ・プロセス

動的に開始される ECP サーバ・プロセスは、“ECP” で始まるルーチン名を持ちます。ユーザ名またはオペレーティング・システム・ユーザ名は通常は **Daemon** または **%System** ですが、Windows 上のインスタンス・サービス・ユーザの名前である場合もあります。以下にプロセス名の例を示します。

- ECPCliR – ECP クライアント・リーダ
- ECPCliW – ECP クライアント・ライター
- ECPSrvR – ECP サーバ・リーダ
- ECPSrvW – ECP サーバ・ライター

Web サーバ・プロセス

Web サーバ・プロセスは動的に開始されます。これらのプロセスは、アイドル状態でタスクを待っているときは、User 列に CSPSystem と表示されます。これらのプロセスがアクティブのときは、Web セッションの InterSystems IRIS ユーザと現在のルーチン名が表示されます。OS Username 列には Web Gateway と表示されます。

- %SYS.cspServer および %SYS.cspServer2 – Web アプリケーション要求を処理するためにプロセスで使用される Web サーバ・ルーチン。
- %SYS.cspServer3 – 非同期通信を処理し、Web ゲートウェイ管理を行うためにプロセスで使用される Web サーバ・ルーチン。

これらのプロセスは、他のインターシステムズ製品の従来のアプリケーションに関連付けられています。このようなアプリケーションにおけるこれらのルーチンの詳細は、この機能に関するよくある質問で該当する質問を参照してください。

注釈　これらのルーチンはライセンスを使用しません。ライセンスは、Web アプリケーション・セッションに関連付けられます。

これらのサーバそれぞれの実行可能ファイルは、Windows では CSPAP.dll であり、UNIX® では CSPap.so です。オペレーティング・システム・ユーザ名は `Web Gateway` です。プログラム名は、プロセスでタスクが変更されるたびに変化する可能性があります。

ミラー・システム・プロセス

ミラー・システム・プロセスが開始されるのは、ミラーリングが構成されている場合です。これらのプロセスは、ミラーリングに関するさまざまな機能を実行します。

- MIRRORMGR – ミラー・マスター。ユーザ名は、実行されるミラー機能を表します (`Mirror Master`、`Mirror Primary`、`Mirror Dejournal`、`Mirror Prefetch`、または `Mirror JrnRead`)。オペレーティング・システム・ユーザ名は `Daemon`です。TCP ポートは開かれません。デバイスはオペレーティング・システムの NULL デバイスです。
- MIRRORCOMM – ミラー通信プロセス。ユーザ名は `Mirror Arbiter`、`Mirror Backup`、または `Mirror Svr:RdDmn` です。オペレーティング・システム・ユーザ名は `Daemon`です。デバイスは `|TCP|XXX`です。TCP ポートは、デバイス名またはミラー構成から確認できます。

IP プロトコル

TCP

InterSystems IRIS インスタンスは、構成オプションで指定されている TCP/IP ポート上の接続を受け付けます。ポートの使用に関するオペレーティング・システム側の制約事項 (ファイアウォールに関するものなど) がある場合は、InterSystems IRIS 向けに構成されているポートと一貫したポート設定によって着信アクセスを許可する必要があります。ファイアウォールで実行可能ファイルのルールが設定されている場合は (Windows 上のファイアウォールでルールが設定されているのと同様に)、必要に応じてプログラムにも許可を付与してください。例えば、irisdb.exe、licmanager.exe、ISCAgent.exe、および httpd.exe の各実行可能ファイルはこのような許可を必要とします。

InterSystems IRIS で使用される TCP/IP ポートは、インスタンス構成によって設定されています。構成されているポートは、インストール・ディレクトリ内の `iris.cpf` ファイルで調べることができます。`[Startup]` セクションでは、`DefaultPort`、`DefaultPortBindAddress`、および `WebServerPort` を構成します。`DefaultPort` では、スーパーサーバが接続を受け付けるポートを指定します。既定値は 1972 です。`DefaultPortBindAddress` では、必要に応じてスーパーサーバのバインド先であるインタフェース・アドレスを指定します。`WebServerPort` では、プライベート Web サーバが接続を受け付けるポートを指定します。既定値は 52773 です。

プライベート Web サーバはほとんどの場合は開発環境で使用されるため、運用環境で使用することは推奨されません。

`[SQL]` セクションにある `JDBCGatewayPort` では、Java Database Connectivity (JDBC) ゲートウェイ・ポート番号を定義します。既定値は 62972 です。

`[Telnet]` セクションにある `Port` 値では、InterSystems Telnet サービス (ctelnetd.exe) が Windows 上の InterSystems IRIS への Telnet 接続を受け付けるポートを指定します。

UDP

InterSystems IRIS とライセンス・サーバ (licmanager または licmanager.exe) は、主に UDP プロトコルを使用して通信します。InterSystems IRIS は、メッセージを UDP パケットとしてライセンス・サーバのポートに送信します。このポートは既定では 4002 であり、管理ポータルの [システム管理]→[ライセンス]→[ライセンスサーバ] で設定します。ライセンス・サーバが InterSystems IRIS に応答するために使用するポートは、InterSystems IRIS が元のメッセージを送信するために使用したポートです (ライセンス・サーバはパケット・ヘッダで該当ポートを確認します)。TCP は、クエリ要求時に InterSystems IRIS とライセンス・サーバの間でのみ使用されます。InterSystems IRIS は、受け付け/リッスンのために TCP ポートをオープンして、このポート番号をクエリ要求に格納して送信します。ライセンス・サーバはそのポートに接続して、結果を TCP

接続を介して送信します。ポート番号はライセンス・サーバのポート番号とします。そのようにしないとポート 0 が使用され、開いているポートを無作為に選択するようにオペレーティング・システムに指定することになります。ここで指定したポートは、クエリ結果の送信時にのみ開きます。

SNMP

%System_Monitor サービスを使用すると、InterSystems IRIS は管理対象システム上の SNMP エージェントに対するサブエージェントとして機能します。このサービスは、InterSystems IRIS の管理とデータの監視 (提供されている MIB で定義) のための SNMP 要求 (GET または GETNEXT) と、SNMP トラップ (InterSystems IRIS によって送信される非同期通知) の両方をサポートしています。%System_Monitor サービスを無効にすると、ローカル・システム上の SNMP エージェントと InterSystems IRIS の間のすべての通信が無効になり、その結果としてすべてのリモート SNMP マネージャ・アプリケーションとの通信も無効になります。

HTTP

HTTP 要求を処理するために InterSystems IRIS で使用される Web ゲートウェイのコンポーネントに関する説明を参照してください。このためには、[ドキュメント]→[InterSystems IRIS Web 開発]→[Web ゲートウェイ・ガイド]→[Web ゲートウェイの概要] の順に選択して、オンライン・ドキュメントにアクセスします。プライベート Web サーバは、インストール・ディレクトリ下の **httpd¥bin** サブディレクトリにある httpd.exe (UNIX® 上では httpd) です。プライベート Web サーバの開始を制御するには、管理ポータルで [システム管理]→[構成]→[追加設定]→[開始] を選択して、[ウェブサーバ] を true または false に設定します。

ゲートウェイ

InterSystems IRIS は、外部データに対するいくつかのゲートウェイを提供しています。これらのゲートウェイには、SQL ゲートウェイ、JDBC ゲートウェイ、オブジェクト・ゲートウェイ、および XSLT 2.0 ゲートウェイのサーバが含まれます。使用される TCP/IP ポートを定義するには、管理ポータルで [システム管理]→[構成]→[接続性] を選択してアクセスできるゲートウェイ・セットアップ・ページを使用します。これらのゲートウェイが依存しているオペレーティング・システムのサービスやプロセスの詳細は、各ゲートウェイのドキュメントを参照してください。

不要な InterSystems IRIS プロセスの削除

InterSystems サービス・プロセスが作成されるのは、これらのプロセスがサポートしているサービスが有効化および構成されている場合のみです。InterSystems サービス・プロセスが実行されることを防止するために、追加の操作を実行する必要はありません。

外部プロセス

InterSystems IRIS インスタンスは、このインスタンスをサポートするいくつかの機能を実行するために、**iris**[.exe] 以外の実行可能ファイルを実行するプロセスを開始します。これらの実行可能ファイルのインスタンス固有バージョンは (これらの実行可能ファイルは通常はインスタンス・バージョンごとに異なります)、インストール・ディレクトリの **bin** サブディレクトリにあります。複数の InterSystems IRIS インスタンスによって共有される可能性のある実行可能ファイルは、共通のディレクトリに格納されています。

以下に、永続プロセスによって実行される可能性のある実行可能ファイルを示します。これらの実行可能ファイルは Windows 上の **bin** ディレクトリに格納されています。

- **irisdb.exe** — InterSystems IRIS 実行可能ファイル。
- **licmanager.exe** — InterSystems IRIS ライセンス・サーバ。
- **CStudio.exe** — スタジオ。
- **iristray.exe** — システム・トレイ内の InterSystems IRIS ランチャー。
- **Iristerm.exe** — ターミナル。
- **iristrmd.exe** — ローカルのターミナル接続デーモン。ローカルのターミナル接続 (Telnet ではなく) を受け付けて、その接続を処理するための InterSystems IRIS サーバ・プロセスを作成します。

- irisirdimj.exe — InterSystems IRIS の始動時とシャットダウン時に WIJ ファイルを処理する実行可能ファイル。

以下に、永続プロセスによって実行される可能性のある実行可能ファイルを示します。これらの実行可能ファイルは UNIX® 上の bin ディレクトリに格納されています。

- iris — InterSystems IRIS 実行可能ファイル。
- licmanager — InterSystems IRIS ライセンス・サーバ。
- irisirdimj — InterSystems IRIS の始動時とシャットダウン時に WIJ ファイルを処理します。

bin ディレクトリ内の他のプログラムもたまに使用されますが、これらのプロセスは短時間しか実行されないため、プロセスのリスト表示で長時間表示されることはあまりありません。

複数の InterSystems IRIS インスタンスによって共有される実行可能バイナリは、Windows 上の C:¥Program Files (x86)¥Common Files¥InterSystems のサブディレクトリに格納されています。これらのプロセスは、これらの実行可能バイナリを Windows 上の共通ディレクトリから実行しているものとして表示されることがあります。

- ISCAgent.exe – ミラーのフェイルオーバーを制御します。
- Iristerm.exe – ターミナル。

共有バイナリは通常、UNIX®上の /usr/local/etc/irissys にインストールされます。

- ISCAgent* – ミラーのフェイルオーバーを制御します。

実行可能バイナリに加えて、いくつかの共有ライブラリ・バイナリが共通ディレクトリに格納されています。

相互運用性

アダプタ

InterSystems IRIS は、アダプタを使用して外部インタフェースとの通信を可能にします。

電子メール

電子メール・アダプタは InterSystems IRIS プロセスです。これらのアダプタは、TCP/IP を使用して電子メール・サーバとの間で電子メールを送受信します。発信アダプタは、SMTP サーバにメールを送信します。着信アダプタは、POP3 サーバからの該当する (フィルタ処理された) メッセージをポーリングします。電子メール・サーバはリモート・サーバ上に配置されている可能性が高いため、ローカル・プロセスは存在しない一方で、リモート・システムにファイアウォールを介してアクセスできる必要があります。

ファイル

ファイル入力アダプタは InterSystems IRIS プロセスです。これらのアダプタは、監視対象として構成されたディレクトリを定期的に調べて、そのディレクトリにあるファイルを読み取って、サポート対象として構成されたビジネス・サービスにそれらのファイルを渡して、構成されたアーカイブ・ディレクトリにそれらのファイルを移動します。EnsLib.File.InboundAdapter クラスは実装を提供します。FilePath、WorkPath、および ArchivePath の各プロパティは、それぞれ入力ディレクトリ、一時作業ディレクトリ、およびアーカイブ・ディレクトリを定義します。

ファイル出力アダプタは、プロダクションのビジネス・オペレーションによってデータをファイルに書き込むために使用されます。ファイルのパスと名前はビジネス・オペレーションによって指定され、ファイルに対する処理は、EnsLib.File.OutboundAdapter クラスのメソッドを呼び出すことで実行されます。メッセージは通常、実際の出力処理を実行するワーカ・ジョブのキューに格納されます。このことは、Ens.Queue プロセスの存在を暗黙的に意味します。

FTP

InterSystems IRIS は、%Net.FtpSession クラスを使用したリモート FTP サーバとの FTP 通信用のクライアントとして機能します。%Net.FtpSession クラスは、着信接続を回避するために、データ・チャンネルに対して PASV を使用するように構成できます。InterSystems IRIS は、FTP の着信アダプタと発信アダプタを提供します。どちらも FTP クライアントとして機

能して、ユーザによって作成されたビジネス・サービスの管理下で get (入力) または put (出力) を実行します。FTP のサーバとポートは構成可能です。FTP アダプタは InterSystems IRIS プロセスです。

HTTP

HTTP アダプタ (EnsLib.HTTP.InboundAdapter および EnsLib.HTTP.OutboundAdapter) は、プロダクションが HTTP 要求と HTTP 応答を送受信することを可能にします。HTTP アダプタは InterSystems IRIS プロセスによって実装されます。着信 HTTP アダプタのポートとインタフェースの IP アドレスは構成可能です。発信 HTTP アダプタの対象であるサーバとポートは、クラス設定によって指定されます。

Java ゲートウェイ

プロダクションのアダプタは、Java ゲートウェイを使用して Java 中間プロセスを介して通信します。Java 仮想マシンの存在に依存する Java プロセスが開始されます。InterSystems IRIS サーバ・プロセスは、TCP 接続を介して Java プロセスと通信します。使用される TCP ポートは構成可能です。

LDAP

ビジネス・サービスは、EnsLib.LDAP.OutboundAdapter クラスを他のアダプタと同じように使用して LDAP サーバに要求を送信したり応答を受信したりできます。

MQSeries

EnsLib.MQSeries.InboundAdapter クラスと EnsLib.MQSeries.OutboundAdapter クラスを使用すると、プロダクションは、IBM WebSphere MQ のメッセージ・キューとの間でメッセージを送受信できます。動的に読み込まれる共有ライブラリ・バイナリが通信用に使用されます。

パイプ

EnsLib.Pipe.InboundAdapter クラスと EnsLib.Pipe.OutboundAdapter クラスを使用すると、プロダクションはオペレーティング・システムのコマンドやシェル・スクリプトを実行できます。これらは、InterSystems IRIS の外部プロセスを作成して、パイプを介してこのプロセスと通信するため、パイプ・アダプタが外部プロセスと通信している間は外部プロセスは存続します。このプロセスによって実行されるコマンドは、アダプタ・クラスの CommandLine プロパティに指定された値によって決まります。

SAP

Java ゲートウェイは、EnlLib.SAP.BootStrap クラスの ImportSAP メソッドを使用してインポートされたクラスを使用して SAP Java コネクタと通信するために使用されます。

SQL

SQL 着信アダプタおよび発信アダプタは、プロダクションが JDBC または ODBC に準拠したデータベースと通信することを可能にします。一般に、着信 SQL アダプタ (EnsLib.SQL.InboundAdapter) はクエリを定期的に実行してから、結果セットの行を繰り返し処理して、関連付けられたビジネス・サービスに 1 行ずつ渡します。SQL アダプタは、InterSystems SQL ゲートウェイと JDBC ゲートウェイの基盤機能を使用します。

TCP

InterSystems IRIS は、入力 TCP アダプタと出力 TCP アダプタを提供します。各 TCP 着信アダプタは、指定されたポート上でデータの有無を確認して、入力を読み取り、関連付けられたビジネス・サービスに入力をストリームとして送信します。プロダクション内では、発信 TCP アダプタは、ユーザによって作成および構成されたビジネス・オペレーションと関連付けられています。このビジネス・オペレーションは、そのプロダクション内からのメッセージを受信して、メッセージ・タイプを調べて、発信 TCP アダプタ内で適切なメソッドを実行して、TCP を介してデータを送信します。

Telnet

InterSystems IRIS が提供する EnsLib.Telnet.OutboundAdapter を使用すると、別のシステム上の telnet 機能への発信 telnet 接続が可能になります。このアダプタが提供するメソッドを使用して、telnet クライアント・ソフトウェアを使用してリモート・システムに手動でログインする機能をプログラムによってエミュレートします。InterSystems IRIS TCP デバイスは基盤テクノロジです。

セキュリティのチェックリスト

導入環境のセキュリティを強化するためのチェックリスト

このチェックリストの目的は、環境のセキュリティ・レベルを検証するためのガイドラインを提示すること、および環境のセキュリティを強化するためのヒントを提示することです。これらのヒントは、組織のセキュリティ侵害を回避および防止するのに役立ちます。このチェックリストは“ハウツー・リスト”として使用されるべきものではなく、全網羅的なものでもありません。以下に示す項目は考慮すべき事項であり、適用すべきルールの絶対的なリストではありません。

インフラストラクチャのセキュリティについて全責任を負っている担当者として、セキュリティ強化と保護のための手法について不安がある場合、セキュリティ専門家にご相談ください。

ネットワークとファイアウォール

ID	トピック	説明
1.	ネットワーク、ハードウェア、ソフトウェア、およびポリシー	セキュリティ・ポリシー、ファイアウォール・ログ、ファイアウォールの構成とパッチ・レベル、公開されている IP アドレス、ネットワーク図、およびファイアウォール・トポロジの情報を取得してレビューします。
2.	物理的環境の監査	ファイアウォールと管理サーバが、許可された人物のみがアクセスできる物理的に安全な場所にあることを確認します。また、それらに最新のパッチが適用されていることを確認します。
3.	変更管理プロセス、ルール・ベース変更のレビュー	変更の手順と承認プロセスをレビューします。このための自動化ツールが提供されています。
4.	脆弱性のテスト	自動化されたツールを実行して、安全性の低いサービス、プロトコル、およびポートを分析して特定します。
5.	総当たり攻撃検知システムの使用	パスワードが不特定の人々に推測されることを阻止して、サーバ・ファイアウォールで不特定の人々の現在の IP アドレスをブロックすることで、サーバに接続することを防止します。
6.	継続的な監査およびリアルタイムの監視とアラート発行	ファイアウォールを継続的に監査するためのプロセスを実行します。ファイアウォールに変更が加えられたときにアラートを発行するためのリアルタイム監視を実行します。これらに関するログを定期的にレビューします。

オペレーティング・システム

ID	トピック	説明
1.	インストール計画	サーバ・ロールを理解して、インストール手順を文書化します。詳細は、オペレーティング・システムの適切なセキュリティ強化ガイドをダウンロードして参照してください。
2.	パッチ・レベル	オペレーティング・システムに最新のパッチが適用されていることを確認します (特にセキュリティ・パッチ)。自動更新を無効にします。
3.	エンドポイント保護ソフトウェア	このソフトウェアをインストールし、適切に構成します(これまでは、ウイルス対策ソフトウェアとしていました)。
4.	不要なソフトウェア、サービス、およびポートの無効化	不要なネットワーク・サービスを無効にします (IPv6、telnet、FTP など)。 使用されていない不要なデーモンを無効にします (DHCP、スケジューリングとキューイングのサービス、ラップトップ・サービスなど)。 使用されているサービスのセキュリティを可能な限り高めます。例えば、SSH プロトコルをバージョン 2 に限定することで SSH のセキュリティを向上させます (バージョン 1 はセキュリティが不十分です)。
5.	ログ	サーバ・ログを保持して、それらのログを別個のログ・サーバにミラーリングします。
6.	監視とアラート発行	監視とアラート発行の設定を通じて、システムに対する変更や未承認アクセスなどのイベントが通知されるようにします。
7.	物理的なセキュリティ	BIOS の構成を通じて、CD/DVD、フロッピー、および外部デバイスから起動できないようにして、これらの設定を保護するためのパスワードを設定します。

Web サーバ

ID	トピック	説明
1.	インストール計画	Web サーバのロールとコンテンツを理解して、ページが静的かどうか、および提供される Web サービスの内容を確認します。インストール手順を文書化します。適切なセキュリティ強化ガイドをダウンロードして参照します。
2.	パッチ・レベル	Web サーバが最新状態であることを確認します (特にセキュリティ・パッチのバージョン)。
3.	Web サーバのヘッダ情報	実行されている Web サーバ・ソフトウェアや、システムのタイプとバージョンに関する情報が HTTP ヘッダに含まれないようにサーバを構成します。
4.	HTTP TRACE の無効化	HTTP TRACE が有効化されているときは、HTTP TRACE 要求を使用してすべての受信情報がエコー・バックされます。
5.	エラー処理	汎用のエラー・ページとエラー処理ロジックを使用して、アプリケーションで既定のエラー・ページを強制的に回避させることで、適切なエラー処理を実現します。既定のエラー・ページは多くの場合、システムとアプリケーションに関する機密情報を漏洩させます。
6.	モジュールの無効化	使用されていないモジュールをすべて無効化することで、Web サーバの外部にさらされる領域を減らします。これらのモジュールによって多くの場合は必要以上の情報が提供されます。 Apache: autoindex、cgi、imap、info、status、userdir、actions、negotiation… IIS: ASP、ASP.NET、WebDAV、CGI、ディレクトリ参照…
7.	ユーザとグループ	Apache: Apache を別個のユーザおよびグループとして実行することで、Apache プロセスを他のシステム・プロセスによって使用できないようにします。 IIS: 使用されていないアカウントを削除して、Guest アカウントを無効にします。

ユーザ、パスワード、グループ、所有権、および権限

ID	トピック	説明
1.	ユーザ管理	root ログインを無効にします。すべての管理者は名前を持つユーザである必要があります。使用されていないユーザ・アカウントがないこと、および既定のユーザ・アカウントとパスワードが使用されていないことを定期的に確認します。
2.	パスワード・ポリシー	大文字と小文字、数字、および特殊文字を組み合わせた非常に強力なパスワードを使用することを義務付けます。 パスワードを定期的に変更します。 ログインの失敗が一定回数を超えた場合は、そのアカウントをロックします。
3.	UNIX®	インストール前にグループとユーザを作成します。 InterSystems IRIS を root としてインストールします。InterSystems IRIS データベースのグループ、所有権、および権限が指定されたとおりに保持されていることを確認します。
4.	Windows	Windows Administrator を使用して InterSystems IRIS をインストールしてから、既定の Windows Administrator アカウントを無効にします。Guest アカウントと Help Assistant アカウントも無効にします。

暗号化 (保管中のデータと伝送中のデータ)

ID	トピック	説明
1.	保管中のデータ	ディスク上に保管されているすべての実運用データが暗号化されていることを確認します。
2.	キー管理	キー管理のポリシーと手順をレビューします。
3.	伝送中のデータ	すべての HTTP データ通信が暗号化されていることを確認します (TLS などを使用)。 すべての TLS 構成で最新バージョンが使用されていることを確認します。

インターシステムズのセキュリティ

ID	トピック	説明
1.	インストール	常に、ロック・ダウン初期セキュリティ設定タイプを指定してインストールします。
2.	認証	ユーザとパスワードを定期的にレビューします。
3.	承認	アプリケーションの要件をレビューします。ロール、リソース、およびサービスを定義します。
4.	監査	監査が有効になっていることを確認します。ログを定期的にレビューします。
5.	サービスの無効化	ECP やミラーリングなどのサービスが使用されていない場合は、それらのサービスを有効にしないでください。
6.	使用されていないデータベースとアプリケーションの削除	USER などの使用されていないデータベースを削除します。

ID およびアクセスの管理

ID およびアクセスの管理の概要

インターシステムズの ID およびアクセスの管理の概要

ID およびアクセスの管理 (IAM) は、コンピュータ・ネットワーク上での ID とアクセス権の管理に焦点を当てたフレームワークです。これには、認証と承認が含まれます。認証は、InterSystems IRIS に接続しようとしているユーザの身元を確認するプロセスです。InterSystems IRIS はいくつかの認証メカニズムをサポートします。認証されたユーザは、InterSystems IRIS と通信し、そのツールとリソースを使用できるようになります。承認は、ユーザがどのデータベース・アセットを使用、表示、または変更できるかを判断するプロセスです。

Kerberos 認証

Kerberos 認証について

Kerberos の背景

最も安全な接続を実現するために、InterSystems IRIS では Kerberos 認証システムをサポートしています。この認証システムは、ユーザの身元を確認するための極めて安全で効果的な手段を提供します。Kerberos は、安全性の低いネットワーク上で認証を実現するために、マサチューセッツ工科大学 (MIT) で開発されたもので、巧妙な攻撃から通信を保護します。この保護システムの最大の特長は、ユーザのパスワードが、暗号化されたものであってもネットワーク上には送出されない点にあります。

Kerberos は、信頼されるサードパーティ・システムと呼ばれるものです。パスワードなどの機密性の高い認証情報はすべて Kerberos サーバに保持され、Kerberos サーバそのものは物理的に安全な場所に設置されます。

Kerberos には以下のような特長もあります。

- 長年にわたる実績 ー Kerberos が初めて開発されたのは 1980 年代後半です。その中心となるアーキテクチャと設計は、数多くのサイトで長年使用されています。また、長年の運用で発見された問題は、継続的な改訂で解決されてきました。
- サポート対象のすべての InterSystems IRIS プラットフォームで使用可能 ーもともと UNIX® 向けに開発された Kerberos は、InterSystems IRIS がサポートするすべての UNIX® 系 OS で使用できます。また、Microsoft 社は Windows 2000 以降の Windows に Kerberos を採用しています(Microsoft .NET フレームワークでは Kerberos を直接サポートしていないので、InterSystems IRIS Managed Provider for .NET でも Kerberos はサポートしていません)。
- 柔軟な設定が可能 ー 異種ネットワークに対応できます。
- 高い拡張性 ー Kerberos プロトコルを使用しているので、鍵配布センター (KDC) との対話処理が最小限で済みます。これにより、大規模なシステム上でこのような対話処理がボトルネックになることを防止できます。
- 高速 ー オープン・ソース製品として、Kerberos は長年にわたって徹底的に検討され、最適化されてきました。

Kerberos 認証の基盤となっているのは、AES 暗号化アルゴリズムです。AES (Advanced Encryption Standard) は、一般公開の下で作成されている、ロイヤルティ不要の対称ブロック暗号化方法で、128 ビット、192 ビット、および 256 ビットのキー・サイズをサポートしています。United States National Institute of Standards and Technology (NIST) により採択されるなど、US Federal Information Processing Standard (FIPS) の一部となっています。

Kerberos の背景は、MIT Kerberos の Web サイトおよび Kerberos に関する入手可能な資料のリストを参照してください。

Kerberos の仕組み

Kerberos モデルには、いくつかのアクターがあります。Kerberos で認証されるさまざまなプログラムと人物を総称して、プリンシパルといいます。Kerberos のシステムは、Kerberos Key Distribution Center (KDC) で管理されます。Windows では、Windows ドメイン・コントローラが KDC の役目を果たしています。KDC は、ユーザがプログラムと対話できるようにユーザにチケットを発行します。これらのプログラムそのものは、サービス・プリンシパルで表現されます。ユーザが認証され、サービス・チケットを受け取ると、そのユーザはプログラムを使用できるようになります。

具体的には、Kerberos 認証は 3 つの独立したトランザクションで構成されます。

1. クライアントは “TGT” (“チケット保証チケット”) と暗号化セッション・キーを受け取ります。
2. クライアントは TGT とセッション・キーを使用して、InterSystems IRIS のサービス・チケットと別の暗号化セッション・キーの両方を取得します。
3. クライアントはサービス・チケットと 2 番目のセッション・キーを使用して、InterSystems IRIS への認証を行い、必要に応じて保護された接続を確立します。

該当する場合に表示される最初のパスワードのプロンプトを除き、この処理はユーザには表示されないようになっています。

InterSystems IRIS による Kerberos の使用法

Kerberos によって環境が適切に保護されるように、Kerberos 認証をサポートしているすべてのインターシステムズのサービスで Kerberos を有効にし、Kerberos 認証をサポートしていないインターシステムズのサービスは無効にする必要があります。この要件に対する例外は、インターシステムズの安全な境界内で動作することを意図したサービス (ECP など) で、これらは Kerberos をサポートしていません。これらのサービスは、外部に対して安全が確保されている環境で使用するように設計されているので、有効化と無効化のみを設定できます。

Kerberos の構成の概要

Kerberos 認証を使用できるように InterSystems IRIS インスタンスを構成するには、以下の手順に従います。

1. Kerberos サービスとして実行されるように InterSystems IRIS が設定されていることを確認します。

   その手順は、InterSystems IRIS サーバのオペレーティング・システムと環境のタイプによって異なります。詳細は、"Kerberos を使用したセキュリティ環境の準備" を参照してください。

2. **[認証/ウェブセッションオプション]** ページ (**[システム管理]→[セキュリティ]→[システム・セキュリティ]→[認証/ウェブセッションオプション]**) で、該当する Kerberos メカニズムを有効にします。

3. InterSystems IRIS との接続に使用するサービスを決定し、それ以外のサービスをすべて無効にします。それぞれの接続ツールで使用されるサービスのリストは、表 "接続ツールおよびそのアクセス・モードとサービス" を参照してください。

4. クライアント・サーバ接続について、サーバで要求する Kerberos 接続のセキュリティ・レベルを指定します。これは、サービスを使用する接続の構成要素となる Kerberos 機能をどのように決定するかということです。詳細は、"接続のセキュリティ・レベルの指定" を参照してください。

5. クライアント・サーバ接続について、クライアント側の設定を実行します。これにより、アプリケーションから、その実行時に必要となる情報にアクセスできるようになります。この情報には以下のものがあります。

   - InterSystems IRIS を表すサービス・プリンシパルの名前
   - 許可された接続のセキュリティ・レベル

   この情報の設定では、Windows 優先サーバの構成などの何らかの構成メカニズムが必要になることがあります。詳細は、"クライアントの設定" を参照してください。

6. 認証プロセスでユーザの認証情報を入手する方法を指定します。この方法には、ユーザの Kerberos 証明書キャッシュをチェックする方法と、Kerberos のパスワードを要求するプロンプトをユーザに示す方法があります。詳細は、"ユーザの認証情報の入手" を参照してください。

7. Web 接続を可能最大限に保護するには、各接続にセキュア・チャンネルを設定します。

重要　Windows では、ドメイン・アカウントを使用してログインする場合、OS ベースの認証と Kerberos 認証は同じものになります。ローカルでログオンする場合、Kerberos は KDC スプーフィング攻撃の標的となるので安全ではなく、お勧めできません。

Kerberos およびアクセス・モードについて

接続ツールごとに、サービスを使用して InterSystems IRIS との接続が設定されます。特定のアクセス・モードも使用されます。最大限の保護を実現するには、使用している接続ツールに基づいて、必要なサービスを決定します。使用しないサービスがあれば無効にします。

以下は、接続ツールおよびそのアクセス・モードとサービスのリストです。

テーブル B-1: 接続ツールおよびそのアクセス・モードとサービス

接続ツール	アクセス・モード	サービス
InterSystems IRIS Telnet	クライアント・サーバ	**%Service_Telnet**
コールイン	ローカル	**%Service_CallIn**
コンソール	ローカル	**%Service_Console**
Java	クライアント・サーバ	**%Service_Bindings**
JDBC	クライアント・サーバ	**%Service_Bindings**
ODBC	クライアント・サーバ	**%Service_Bindings**
ターミナル	ローカル	**%Service_Terminal**
Web テクノロジ	Web	**%Service_WebGateway**

ローカル

ローカル・サービスの Kerberos 認証では、ユーザと InterSystems IRIS の両方が有効な Kerberos プリンシパルとして設定されます。この場合、使用されているマシンは 1 台のみで、そのマシン上でプロセスが 1 つのみ実行されています。したがって、ポータルにあるこれらのサービスの構成ページを使用すると、Kerberos プロンプト (管理ポータルでは単に Kerberos とラベル表示されています)、または Kerberos 証明書キャッシュのどちらを使用するかを指定できます。

このシナリオでは、ユーザと InterSystems IRIS は同じマシン上で同じプロセスを使用しているので、両者の間に接続は存在しません。両者はプロセスを共有していることから、安全でない媒体を通じて情報が受け渡されることがなく、したがって両者のデータに特別な保護を設ける必要はありません(この状況をプロセス内認証といいます)。

クライアント・サーバ

クライアント・サーバ・アプリケーションでは、Java、JDBC、ODBC、および Telnet からの接続を扱います。Kerberos 認証を使用するクライアント・サーバ・アプリケーションを介して InterSystems IRIS を操作するユーザには、認証情報が必要です。

サーバとクライアントのそれぞれを構成する必要があります。サーバの構成では、受け入れる接続のタイプを指定します。クライアントの構成では、使用する接続のタイプを指定します。また、ユーザの資格情報を入手する方法も指定できます。

クライアント・サーバ接続では、接続のさまざまなセキュリティ・レベルが Kerberos でサポートされていて、これらのセキュリティ・レベルは InterSystems IRIS サーバ・マシン上で構成します。このレベルには、以下のものがあります。

- Kerberos ー ユーザと InterSystems IRIS との間の最初の認証が Kerberos で管理されます。それ以降の通信は保護されません。
- Kerberos パケット整合性 ー ユーザと InterSystems IRIS との間の最初の認証が Kerberos で管理されます。それ以降取り交わされる各メッセージは、ソースとコンテンツの検証を可能にするハッシュを備えています。これにより、各方向で取り交わされる各メッセージが、そこに示されている送信者から本当に送信されたものであることを検証できます。また、送信者から受信者への伝送の過程でメッセージが改ざんされていないことも検証できます。
- Kerberos 暗号化 ー Kerberos によって最初の認証が管理され、すべての通信の整合性が確保されます。また、暗号化も Kerberos で実行されます。これには、ユーザと InterSystems IRIS の間でやり取りされるすべてのメッセージを各方向でエンド・ツー・エンド暗号化する処理も含まれます。

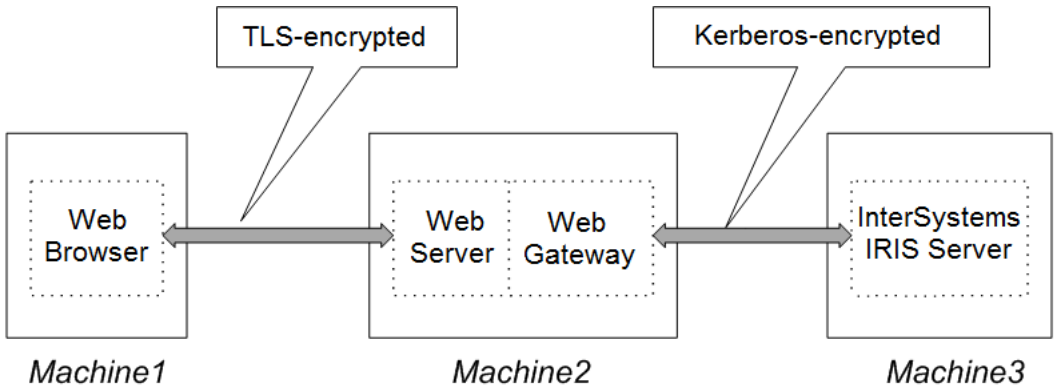
Web

Web アプリケーションの実行では、ユーザと InterSystems IRIS サーバとの間に直接の対話は発生しません。監視からすべての情報を保護するには、ユーザと InterSystems IRIS との間の接続を以下のように暗号化する必要があります。

- TLS を使用してブラウザから Web サーバへの接続を保護するよう Web サーバを構成します。
- Web サーバと Web ゲートウェイは同じ場所に置かれるため、両者間の接続を保護する必要はありません。これが不可能な場合は、他の方法を通じて接続上のデータの安全が確保されるようにしてください。
- Kerberos 認証と暗号化を使用するように Web ゲートウェイを構成します。このような接続を確立するには、ゲートウェイの Kerberos プリンシパルを使用します。

この場合のアーキテクチャは以下のとおりです。

図 B-1: Kerberos で保護された Web 接続のアーキテクチャ



エンドユーザと InterSystems IRIS との間の通信は、すべて TLS で暗号化されたパイプまたは Kerberos で暗号化されたパイプを通じて行われます。Kerberos で保護された接続の場合、この通信にはエンドユーザの Kerberos 認証が含まれます。

エンドユーザにパスワードを要求するプロンプトを InterSystems IRIS サーバから表示することはできないので、プロンプトを表示する HTML コンテンツをブラウザに送信する API が呼び出されます。送信されたフォームにユーザが所要の情報を入力すると、その情報は Web サーバに返送されます。この情報は、Web サーバから Web ゲートウェイに渡され、さらにそこから InterSystems IRIS サーバに渡されます。Web サーバは、ブラウザを使用しているユーザのためにプロキシとして機能します。この種類の接続がプロキシ接続と呼ばれるのはこのためです。同時に、ローカル・アクセス・モード同様、ユーザに関連するすべての情報はサーバ・マシンに存在します。したがって、Web 接続は、プロセス内認証の 1 つの形式ともいえます。

接続のセキュリティ・レベルの指定

InterSystems IRIS とのクライアント・サーバ接続では、以下のいずれかのサービスが使用されます。

- **%Service_Bindings** — Java、JDBC、ODBC
- **%Service_Telnet** — Telnet

これらのサービスのいずれかを使用している Kerberos 接続では、その接続をサーバが受け入れるためのセキュリティ・レベルを指定する必要があります。目的のサービスでサポートされる接続のセキュリティ・レベルを構成するには、以下の手順に従います。

1. **[認証/Web セッション・オプション]** ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[認証/Web セッション・オプション]**) で、InterSystems IRIS インスタンス全体に対して有効にする接続セキュリティ・レベルを指定します。以下のレベルにできます。
   - **[Kerberos]** — 最初の認証のみ
   - **[Kerberos パケット整合性]** — 最初の認証およびパケットの整合性
   - **[Kerberos 暗号化]** — 最初の認証、パケットの整合性、およびすべてのメッセージの暗号化

[**認証オプション**] ページの詳細は、"認証オプション" を参照してください。

2. [**サービス**] ページ ([**システム管理**]→[**セキュリティ**]→[**サービス**]) の [**名前**] 列でサービス名をクリックすると、そのサービスの [**サービス編集**] ページが表示されます。
3. [**サービス編集**] ページで、Kerberos 接続の一部として要求する接続のセキュリティ・レベルを指定します。レベルを選択したら、[**保存**] をクリックします。

サーバに指定されているセキュリティ・レベルより低いレベルのセキュリティを使用して、クライアントがそのサーバに接続しようとしても、その接続は受け入れられません。サーバに指定されているセキュリティ・レベルより高いレベルのセキュリティを使用してクライアントがそのサーバに接続しようとすると、そのサーバ接続では、サーバに指定されたレベルのセキュリティを使用して認証が試行されます。

クライアントの設定

クライアント・サーバ・アクセス・モードを使用する場合は、クライアントを構成する必要があります。このプロセスの詳細は、使用している接続テクノロジによって異なります。

Telnet : Kerberos で使用する優先接続サーバの設定

Windows クライアントの場合、Windows 用の InterSystems IRIS Telnet を使用して接続を確立するには、リモート・サーバの一部として格納されている構成情報を使用します。

重要　InterSystems IRIS は、Windows 用に専用の Telnet サーバを備えています。Windows 以外で稼動しているマシンに接続しても InterSystems IRIS Telnet サーバは使用できません。この場合は、オペレーティング・システム付属の Telnet サーバを使用します。サーバ・マシンとの接続が確立されれば、**%Service_Terminal** サービスを使用して InterSystems IRIS を起動できます。

Telnet によるクライアント接続を構成するには、該当のクライアント・マシンに移動します。そのマシンで以下の手順を実行します。

1. InterSystems IRIS ランチャーをクリックし、メニューから [**優先接続サーバ**] を選択します ([**優先接続サーバ**] の選択対象には、現在の優先接続サーバの名前も表示されています)。
2. 表示されたサブメニューから、[**追加/編集**] を選択します。
3. リモート・サーバを新規に作成するには、[**追加**] ボタンをクリックします。既存のサーバを構成するには、接続先の InterSystems IRIS サーバを選択し、[**編集**] ボタンをクリックします。
4. [**接続を追加**] ダイアログが表示されます。このダイアログの [**認証方法**] 領域で [Kerberos] をクリックします。ダイアログが広がり、追加のフィールドがいくつか表示されます。
5. 既存のサーバの値を編集する場合は、このダイアログにある一般的な内容のフィールドで値を変更したり、追加したりする必要はありません。これらの値は、編集するサーバを選択することで自動的に決まります。

   新規サーバを追加する場合に入力するフィールドの詳細は、"リモート・サーバ接続の定義" を参照してください。
6. このダイアログの Kerberos に関するフィールドで、以下のフィールドの値を指定します。
   - 接続セキュリティレベル。Kerberos 認証のみ、Kerberos 認証とパケット整合性、または Kerberos 認証、パケット整合性、および暗号化 を選択できます。
   - サービスプリンシパル名。サービス・プリンシパル名の設定の詳細は、"名前および名前付け規約" を参照してください。
   - Windows マシンへの Telnet 接続を構成する場合は、接続に Windows InterSystems IRIS Telnet サーバを使用することを指定するボックスにチェックを付けます。
7. [OK] をクリックすると、指定した値が保存され、ダイアログが閉じます。

Kerberos で使用する ODBC DSN の設定

InterSystems IRIS では、Windows、UNIX®、または Mac で稼動しているクライアントから DSN (データ・ソース・ノード) への Kerberos で保護された ODBC 接続を、すべてのプラットフォームでサポートしています。クライアントの動作を構成する手順は、以下のようにプラットフォームによって異なります。

- すべてのプラットフォームで、名前と値の組み合わせのセットを受け取る SQLDriverConnect 関数を使用できます。SQLDriverConnect は ODBC API を構成する C 呼び出しで、その説明は Microsoft の Web サイトで入手できます。この名前と値の組み合わせは、Windows 以外のプラットフォームで使用できる初期化ファイルにあるものと同じです。
- Windows 以外のプラットフォームでは、InterSystems ODBC 初期化ファイルを使用して、接続情報を提供する名前と値の対を指定します。このファイルについては、"InterSystems ODBC ドライバの使用法" で概要を説明しています。ODBC 初期化ファイルには、以下の Kerberos 関連の変数があります。
  - Authentication Method ー ODBC クライアントを DSN に認証する方法を指定します。0 はインスタンス認証、1 は Kerberos を指定します。
  - Security Level ー Kerberos 接続で、接続の保護に使用する機能を指定します。1 は認証のみに Kerberos を使用すること、2 は認証およびクライアントとサーバ間で受け渡されるすべてのパケットの整合性の確保に Kerberos を使用すること、3 は認証、パケットの整合性、およびすべてのメッセージの暗号化に Kerberos を使用することをそれぞれ指定します。
  - Service Principal Name ー DSN として機能しているインターシステムズのサービスの名前を指定します。例えば、サービス・プリンシパルは “iris/localhost.domain.com” という名前を持ちます。

  これらの変数の名前では、各単語の間にスペースを記述する必要があります。大文字と小文字は区別されません。
- Windows クライアントでは、GUI (ODBC DSN 構成ダイアログ) を使って接続情報を指定できます。**[システム DSN]** タブにオプションが表示されます。この画面には、それぞれのフィールドを説明するヘルプがあります。Windows の [スタート] メニューからの操作でこの画面を表示する方法は、Windows のバージョンによって異なりますが、多くの場合は **[管理ツール]** の中から選択できます。

  重要　64 ビットの Windows の場合、**odbcad32.exe** には 2 つのバージョンがあります。1 つは **C:¥Windows¥System32¥** ディレクトリにあり、もう 1 つは **C:¥Windows¥SysWOW64¥** ディレクトリにあります。64 ビットの Windows を実行している場合は、**C:¥Windows¥SysWOW64¥** にあるものを使用して DSN を構成してください。

Kerberos で使用する Java クライアントまたは JDBC クライアントの設定

InterSystems IRIS には、Java クライアントの構成を支援するユーティリティとして機能する Java クラスが用意されています。クライアントを構成する準備ができた段階で、このクラスを実行します。以下はその方法です。

1. Kerberos で使用するクライアントを構成するには、以下のように Java の Configure コマンドを発行します。

   ```
   java -classpath '$IRIS_INSTALL_DIRECTORY/dev/java/lib/JDK18/*' com.intersystems.jgss.Configure
   ```

   これにより、JDK ディレクトリ内からだけでなく、マシン上の任意の場所から Configure を実行できるようになります。このコマンドの詳細は、Windows のパス・スタイルと JDK11 の使用のどちらに対応しているかなど、サイトによって異なることに留意してください。

   このプログラムは、Java Generic Security Services (JGSS) を使用して以下のアクションを実行します。

   - 必要に応じて、**java.security** ファイルを変更します。
   - **isclogin.conf** ファイルを作成または変更します。

2. 次にこのプログラムでは、krb5.conf ファイルを作成して構成することを求めるプロンプトが表示されます。このファイルが存在する場合は、その既存の krb5.conf を使用するか、新しいファイルと置き換えるか選択することを求められます。ファイルを置き換える場合は、以下の情報を求められます。

   a. Kerberos レルム ードメインの既定値として、ローカル・ドメイン名が小文字で提示されます。

   b. プライマリ KDC ー ローカル・マシン名のみを指定すれば、それに Kerberos レルムの名前が自動的に付加されます。

   c. セカンダリ KDC ー ゼロ個以上の KDC の名前を指定して、それにプライマリ KDC の内容を複製できます。

3. これらの情報を入力した後、コマンドをもう一度実行します(実行するように指示されます)。

4. krb5.conf の置き換えを求められたら、既存のファイルをそのまま残すことを選択します。指定した Kerberos レルムにあるプリンシパルのユーザ名とパスワードが要求されるので、指示どおりに入力します。これによって、接続がテストされます。

テストが正常に終了すれば、クライアントの構成は完了です。

ユーザの認証情報の入手

すべてのアクセス・モードについて、アプリケーションでユーザの認証情報を入手する方法を指定する必要があります。この方法には、既存の認証情報キャッシュから得る方法と、ユーザにユーザ名とパスワードを要求して得る方法があります。

ローカル・アクセス・モード用の認証情報の入手

ローカル・アクセス・モードでは、InterSystems IRIS と同じマシンにユーザの認証情報が存在します。この状況では、InterSystems IRIS に接続するサービスがアプリケーションで使用されています。このサービスには、以下のものがあります。

- **%Service_CallIn**
- **%Service_Console**
- **%Service_Terminal**

資格情報の入手方法を指定するには、以下の手順に従います。

1. **[サービス]** ページ (**[システム管理]** > **[セキュリティ]** > **[サービス]**) で、**[名前]** 列からサービスを選択します。選択したサービスの **[サービス編集]** ページが表示されます。

2. **[サービス編集]** ページで、資格情報の入手方法を指定します。プロンプトを表示してユーザに要求する方法 (**[Kerberos]** チェック・ボックス) または資格情報キャッシュを使用する方法 (**[Kerberos 証明書キャッシュ]** チェック・ボックス) を選択します。両方を選択しないでください。

   [保存] をクリックすると、選択した設定内容が使用されます。

注釈　Kerberos (プロンプト) と Kerberos 証明書キャッシュを使用する方法の両方をサービスに対して有効にすると、資格情報キャッシュによる認証が優先されます。この動作は InterSystems IRIS ではなく Kerberos によって指定されたものです。

ドメイン・コントローラを備えた Windows (Windows で多く使用される構成) では、ログインによって Kerberos 証明書キャッシュが設定されます。UNIX®、Linux、および macOS の既定の状態では、通常、Kerberos 資格情報が存在しません。したがって、Kerberos プロンプトを使用するように InterSystems IRIS を構成します。これらのシステムでは、ユーザは以下のいずれかの方法で資格情報を入手できます。

- ターミナルを起動する前に kinit を実行する。
- ユーザに対してログイン・プロセスで Kerberos 認証が実行される条件で、システムにログインする。

これらの状況では、資格情報キャッシュを使用するように InterSystems IRIS を構成できます。

クライアント・サーバ・アクセス・モード用の認証情報の入手

クライアント・サーバ・アクセス・モードでは、クライアント・アプリケーションをホストしているマシンにユーザの認証情報が存在します。この場合は、クライアントが以下のいずれの接続方法を使用しているかによって、認証情報の入手方法を指定する形式が異なります。

- ODBC および Telnet
- Java および JDBC

ODBC および Telnet

これらの接続ツールで使用される基盤の InterSystems IRIS コードでは、エンドユーザが既に資格情報を持っていることが前提となっています。したがって、プロンプトの表示は不要です。

Windows では、ドメインにログオンしているすべてのユーザごとに資格情報キャッシュがあります。

Windows 以外のオペレーティング・システムでは、そのユーザに対してオペレーティング・システムで Kerberos 認証を実行済みである場合、またはユーザが明示的に kinit を実行済みである場合に、そのユーザの認証情報キャッシュが存在します。それ以外の場合は、キャッシュにユーザの資格情報が存在しないので、接続ツールは認証に失敗します。

注釈　オペレーティング・システムによっては、一部の接続ツールが使用できないことがあります。

Java および JDBC

Java および JDBC を使用する際には、2 つの異なる Java 実装 (Oracle による実装と IBM による実装) があります。2 つの実装に共通する動作もあれば、互いに異なる動作もあります。

注釈　Java の IBM による実装は、バージョン 8 の場合にのみ利用可能です。これ以降のバージョンの場合、IBM ではオープン・ソース・バージョンがサポートされます。

どちらの実装でも、java.util.Properties クラスのインスタンスのプロパティに接続情報が格納されます。以下のプロパティがあります。

- user ー InterSystems IRIS サーバに接続しているユーザの名前。この値は特定の接続動作に対してのみ設定されます。
- password ー そのユーザのパスワード。この値は特定の接続動作に対してのみ設定されます。
- service principal name ー InterSystems IRIS サーバの Kerberos プリンシパル名。この値はすべての接続動作に対して設定されます。
- connection security level ー この接続に対して Kerberos が提供する保護のタイプ。1 は認証のみに Kerberos を使用すること、2 は認証およびクライアントとサーバ間で受け渡されるすべてのパケットの整合性の確保に Kerberos を使用すること、3 は認証、パケットの整合性、およびすべてのメッセージの暗号化に Kerberos を使用することをそれぞれ指定します。この値はすべての接続動作に対して設定されます。

以下の説明では、java.util.Properties クラスのインスタンスを connection_properties オブジェクトとして指定しています。各プロパティの値は、次のような connection_properties.put メソッドの呼び出しによって設定されます。

```
String principalName = "MyServer";
connection_properties.put("service principal name",principalName);
```

どちらの実装でも、認証情報に関連する動作は isclogin.conf ファイル内の特定のパラメータの値で決まります (このファイルの詳細は、"Kerberos で使用する Java クライアントまたは JDBC クライアントの設定" を参照)。

2 つの Java 実装の動作には、次の 2 つの相違点があります。

- 認証情報に関連する動作を指定するために isclogin.conf ファイルに設定されるパラメータの名前が実装間で異なります。
  - IBM では、パラメータ名は useDefaultCcache です。
  - Oracle では、パラメータ名は useTicketCache です。
- 使用できる動作が実装間で異なります。これらは以下のセクションで説明しています。

IBM の実装を使用しているクライアントに対する動作の指定

選択項目は、以下のとおりです。

- 認証情報のキャッシュを使用するには、useDefaultCcache パラメータの値を True に設定し、user プロパティと password プロパティには値を設定しません。資格情報キャッシュが使用できない場合は、例外が発生します。
- プログラムによる処理で渡されるユーザ名とパスワードを使用するには、useDefaultCcache パラメータの値を False に設定し、user プロパティと password プロパティに該当の値を設定します。
- ユーザにユーザ名とパスワードを要求するには、useDefaultCcache パラメータの値を False に設定し、user プロパティと password プロパティには値を設定しません。これらのプロパティに値を設定しないことにより、InterSystems IRIS に付属するライブラリのクラスを使用して、ユーザ名とパスワードを要求するプロンプトを生成できます。

Oracle の実装を使用しているクライアントに対する動作の指定

選択項目は、以下のとおりです。

- プログラムによる処理で渡されるユーザ名とパスワードを排他的に使用するには、useTicketCache パラメータの値を False に設定し、user プロパティと password プロパティに該当の値を設定します。
- ユーザにユーザ名とパスワードを排他的に要求するには、useTicketCache パラメータの値を False に設定し、user プロパティと password プロパティには値を設定しません。これらのプロパティに値を設定しないことにより、InterSystems IRIS に付属するライブラリのクラスを使用して、ユーザ名とパスワードを要求するプロンプトを生成できます。
- 認証情報キャッシュを排他的に使用するには、useTicketCache パラメータの値を True に設定します。余分なアクションが発生しないようにするには、user プロパティと password プロパティに、存在しない値を設定します。これによってプロンプトが表示されなくなり、このプロパティの値に基づいて認証しようとしても必ず失敗するようになります。
- 認証情報キャッシュの使用を試し、それが使用できなかった場合にプログラムによる処理で渡されるユーザ名とパスワードを使用するには、useTicketCache パラメータの値を True に設定し、user プロパティと password プロパティに該当の値を設定します。資格情報キャッシュが存在しない場合は、これらのプロパティの値が使用されます。
- 認証情報キャッシュを試し、それが使用できなかった場合にユーザ名とパスワードを要求するには、useTicketCache パラメータを True に設定し、user プロパティと password プロパティには値を設定しません。資格情報キャッシュが存在しない場合は、InterSystems IRIS に付属するライブラリのクラスを使用して、ユーザ名とパスワードを要求するプロンプトを生成できます。

Web アクセス・モード用の認証情報の入手

Kerberos を使用する Web ベース接続では、ユーザ名とパスワードを要求するプロンプトが必ず表示されます。このプロンプトを通じて認証されれば、ユーザ名とパスワードがメモリに置かれ、不要になった時点で破棄されます。

Web 接続で使用するセキュア・チャンネルの設定

Web 接続に最大限の保護を実現するには、2 つの重要な通信経路であるブラウザと Web サーバの間および Web ゲートウェイと InterSystems IRIS の間でセキュア・チャンネルを使用することをお勧めします。これにより、Kerberos のユーザ名とパスワードなどのあらゆる情報が、複数のポイント間で伝送されるときに保護されます。以下の各通信チャンネルを保護するには、以下の手順に従います。

- Web ブラウザと Web サーバの間

- Web ゲートウェイと InterSystems IRIS の間

Web ブラウザと Web サーバ間の接続の保護

Web ブラウザと Web サーバ間の接続を保護するための一般的な手段は、TLS (Transport Layer Security) を使用することです。

Web ゲートウェイと InterSystems IRIS 間での Kerberos で保護された接続の設定

Web ゲートウェイと InterSystems IRIS サーバの間に暗号化されたセキュア・チャンネルを設定するには、このゲートウェイを表す Kerberos プリンシパルが必要です。このプリンシパルによって InterSystems IRIS への暗号化接続が確立され、すべての情報はこの接続を通じて伝送されます。こうすることで、エンドユーザを InterSystems IRIS に認証でき、このプロセスの間でデータ盗用が防止されます。

注釈　TLS で保護された Web ゲートウェイと InterSystems IRIS サーバの間の接続設定に関する詳細は、"TLS を使用して InterSystems IRIS に接続するための Web ゲートウェイの構成" を参照してください。

以下はその方法です。

1. ゲートウェイを表す Kerberos プリンシパルの名前を決定または選択します。

   Windows の場合は、ゲートウェイ・ホストのネットワーク・サービス・セッションを表すプリンシパル名が、この Kerberos プリンシパル名になります (つまり、ゲートウェイをホストしているマシンの名前に "$" を付加した machine_name$ という形式で、例えば Athens$ となります)。Windows 以外のプラットフォームの場合は、ゲートウェイの構成画面でユーザ名として入力した任意の有効なプリンシパル名が、この Kerberos プリンシパル名になります。このプリンシパル名で、キー・テーブル・ファイルにある適切なキーが特定されます。

2. ゲートウェイの Kerberos プリンシパルと同じ名前を持つユーザを InterSystems IRIS に作成します。これを行うには、"ユーザ・アカウントの作成" の手順に従います。

3. 必要な任意のリソースに対する使用、読み取り、または書き込みの許可 (特権ともいいます) をこのユーザに与えます。これらの特権をロールに関連付け、次にユーザをこのロールに関連付けることで、ユーザに特権を与えることができます。

4. **%Service_WebGateway** サービスを構成します。これを行うには、"サービスのプロパティ" で説明しているフィールドに入力します。

5. ゲートウェイからサーバにアクセスできるようにゲートウェイを構成します。以下はその方法です。

   a. 管理ポータルのホーム・ページで、**[ウェブゲートウェイ管理]** ページ (**[システム管理]** > **[構成]** > **[ウェブゲートウェイ管理]**) に移動します。

   b. [ウェブゲートウェイ管理] ページでは、左側に選択項目のセットが表示されています。**[構成]** の下にある **[サーバ接続]** をクリックします。**[サーバ接続]** ページが表示されます。

   c. **[サーバ接続]** ページでは、新規の構成の追加、または既存の構成の編集が可能です。新規の構成を追加するには、**[サーバ追加]** ボタンをクリックします。既存の構成を編集するには、左側のリストから目的の構成を選択し、**[サーバ編集]** ラジオ・ボタンを選択して **[実行]** をクリックします。サーバ接続パラメータを編集または構成するためのページが表示されます。ヘルプ画面に説明がある一般的なパラメータに加え、このページでは、ゲートウェイのセキュリティに関するパラメータを指定できます。Kerberos 接続の場合は、以下のパラメータがあります。

      - **[接続セキュリティレベル]** – この接続に対して Kerberos で提供する保護の種類を選択します。(前の手順で Web サービスに対して指定したセキュリティのタイプ以上のセキュリティ・レベルを選択する必要があります。)
      - **[ユーザ名]** – ゲートウェイを表す Kerberos プリンシパルの名前。(このプロセスの最初の手順で使用したプリンシパル名と同じ名前にする必要があります。)

- **[パスワード]** – これには値を指定しないでください(このフィールドは、インスタンス認証で使用するためにゲートウェイを構成するときに使用します)。
- **[プロダクト]** – InterSystems IRIS。
- **[サービスプリンシパル名]** – InterSystems IRIS サーバを表すプリンシパルの名前。これには通常、“iris/machine.domain”という形式で、標準の Kerberos プリンシパル名を指定します。`iris` は InterSystems IRIS のサービスであることを示す固定文字列、machine はマシン名、domain は“intersystems.com”のようなドメイン名です。
- **[キーテーブル]** – Windows で InterSystems IRIS のインスタンスに接続する場合は、このフィールドを空白のままにしておきます。Windows 以外のオペレーティング・システムの場合は、Web ゲートウェイに属する永続キーを収めたキータブ・ファイルの名前をフル・パスで指定します。

これらの値をすべて入力した後、**[設定を保存]** ボタンをクリックすると、入力した値が保存されます。

これで、Web サービスを構成できるようになります。これは、Web アプリケーションのサポートに必要な基盤となるインフラストラクチャを、CSP サービスで提供できるということです。

保護された Web アプリケーションを作成する場合、アプリケーション開発者は以下の処理を行う必要があります。

1. 認証方法を選択する。
2. アプリケーションのロールを構成する。
3. ブラウザと Web サーバ間の接続に TLS が使用されていることを確認する。

オペレーティング・システム・ベースの認証

OS ベースの認証について

InterSystems IRIS では、オペレーティング・システム・ベース (OS ベース) の認証がサポートされます。オペレーティング・システム認証では、オペレーティング・システムのユーザ ID を使用して、InterSystems IRIS のユーザを識別します。オペレーティング・システム認証が有効になっている場合、ユーザは、オペレーティング・システムのプロトコルに従って、オペレーティング・システムに対する認証を実行します。例えば、UNIX® でネイティブの機能に相当するものは、従来からあるログイン・プロンプトです。このプロンプトでは、パスワードのハッシュ値が、/etc/passwd ファイルに格納されている値とオペレーティング・システム・レベルで比較されます。ユーザが初めて InterSystems IRIS に接続しようとすると、プロセスのオペレーティング・システム・レベルのユーザ識別情報が取得されます。このユーザ識別情報が InterSystems IRIS のユーザ名と一致すれば、そのユーザは認証されます。

この機能は、サーバ側で実行されるプロセスにのみ適用されます。例えば、ターミナル・ベースのアプリケーション (ターミナルによる接続など) やオペレーティング・システムによって起動するバッチ・プロセスなどです。 他のマシンから InterSystems IRIS に接続するアプリケーションでは、この機能は使用できません。例えば、あるマシン上のスタジオが、別のマシン上の InterSystems IRIS サーバに接続する場合です。

この機能は、通常、UNIX® システムで利用されていますが、Windows コンソールでも使用できます。

OS ベースの認証は、ローカル・プロセスに対してのみ使用できます。具体的には、以下のとおりです。

- コールイン (**%Service_Callin**)
- コンソール (**%Service_Console**)
- ターミナル (**%Service_Terminal**)

OS ベースの認証の構成

このタイプの認証の使用を設定するには、以下の手順に従います。

1. **[認証/Web セッション・オプション]** ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[認証/Web セッション・オプション]**) ページで、**[オペレーティングシステム認証を許可]** を選択します。
2. **[サービス]** ページ (**[システム管理]** > **[セキュリティ]** > **[サービス]**) で、**[名前]** 列からサービスを選択します。選択したサービスの **[サービス編集]** ページが表示されます。
3. **[サービス編集]** ページで、**[オペレーティングシステム]** チェック・ボックスにチェックを付けてオペレーティング・システム・ベースの認証を選択します。

   [保存] をクリックすると、選択した設定内容が使用されます。

このタイプの認証では、これ以上のアクションは不要です。

注釈　Windows では、ドメイン・アカウントを使用してログインする場合、OS ベースの認証と Kerberos 認証は同じものになります。

%Service_Console に関するメモ

コンソール (**%Service_Console**) は Windows ベースのサービスであり、Windows ドメインのログインでは Kerberos を使用することが普通なので、コンソールの OS ベース認証でローカル・ログインに対する認証が得られます。

%Service_Callin に関するメモ

コールイン (**%Service_Callin**) では、OS レベルのプロンプトからのみ OS ベース認証を使用できます。プログラム処理でコールインを使用する場合は、OS ベース認証がサポートされていません。この場合は、認証されていないアクセスのみが可能です。

インスタンス認証

インスタンス認証について

InterSystems IRIS 自体で、インスタンス認証と呼ばれるログイン・メカニズムを提供できます(管理ポータルでは、**パスワード認証**と呼びます)。具体的には、ユーザ・アカウントごとにパスワードの値が InterSystems IRIS に保持されており、ログインするユーザが入力した値とその値が比較されます。従来の OS ベースの認証同様、InterSystems IRIS でも、パスワードをハッシュ化した値が格納されています。ユーザがログインすると、入力されたパスワードがハッシュ化され、これら 2 つのハッシュ値が比較されます。システム管理者は、パスワードの最小長などの一定のパスワード基準を設定することで、ユーザが選択するパスワードに所定の堅牢性を確保できます。この基準は、"パスワードの強固さとパスワードのポリシー" で説明されています。

InterSystems IRIS は、パスワードの回復不可能な暗号化ハッシュのみを格納します。ハッシュは、Public Key Cryptography Standard #5 v2.1 "Password-Based Cryptography Standard" の定義に従って、HMAC-SHA-512 擬似ランダム関数と PBKDF2 アルゴリズムを使用して計算されます。 現在の実装では 10,000 回の反復、64 ビットのソルトを使用して、64 バイトのハッシュ値が生成されます。別のアルゴリズムを指定するか、反復回数を増やすには、それぞれ Security.System.PasswordHashAlgorithm メソッドと Security.System.PasswordHashWorkFactor メソッドを使用します。これらのハッシュ値から元のパスワードを回復するための既知のテクニックはありません。

インスタンス認証による認証で使用できるサービスは、以下のとおりです。

- **%Service_Binding**
- **%Service_CallIn**
- **%Service_ComPort**
- **%Service_Console**
- **%Service_Telnet**
- **%Service_Terminal**
- **%Service_WebGateway**

インスタンス認証の構成の概要

インスタンス認証を使用するサービスは、以下のように構成する必要があります。

1. **[認証/ウェブセッションオプション]** ページ (**[システム管理]→[セキュリティ]→[システム・セキュリティ]→[認証/ウェブセッションオプション]**) で、**[パスワード認証を許可]** を選択して、インスタンス認証を使用した認証を有効にします。
2. 特定のサービスについては、**[サービス]** ページ (**[システム管理]→[セキュリティ]→[サービス]**) に移動し、**[名前]** 列で [%Service_Bindings] などの目的のサービスを選択します。そのサービスの **[サービス編集]** ページが表示されます。
3. このページでインスタンス認証を選択します。インスタンス認証は、認証タイプのリストで単に **[パスワード]** と表示されています。
4. **[保存]** をクリックすると、選択した設定内容が保存されます。
5. この基本的な手順に加え、一部のサービスではさらに構成が必要です。これは、以下のセクションで説明します。
   - Web
   - ODBC
   - Telnet

Web

Web アクセスでは、必要に応じて、インスタンス認証を通じ、Web ゲートウェイでゲートウェイ自身を InterSystems IRIS サーバに認証するように要求できます。この構成を実行するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、**[ウェブゲートウェイ管理]** ページ (**[システム管理]** > **[構成]** > **[ウェブゲートウェイ管理]**) に移動します。
2. [ウェブゲートウェイ管理] ページでは、左側に選択項目のセットが表示されています。**[構成]** の下にある **[サーバ接続]** をクリックします。**[サーバ接続]** ページが表示されます。
3. **[サーバ接続]** ページでは、新規の構成の追加、または既存の構成の編集が可能です。新規の構成を追加するには、**[サーバ追加]** ボタンをクリックします。既存の構成を編集するには、左側のリストから目的の構成を選択し、**[サーバ編集]** ラジオ・ボタンを選択して **[実行]** をクリックします。サーバ接続パラメータを編集または構成するためのページが表示されます。ヘルプ画面に説明がある一般的なパラメータに加え、このページでは、ゲートウェイのセキュリティに関するパラメータを指定できます。インスタンス認証接続の場合は、以下のパラメータがあります。
   - **[接続セキュリティレベル]** – **[パスワード]** をドロップダウン・リストから選択してインスタンス認証を使用します。
   - **[ユーザ名]** – ゲートウェイ・サービスが実行されるユーザ名 (インストール・プロセスでは、この目的のために CSPSystem ユーザが作成されます)。このユーザ (CSPSystem またはその他のユーザ) には、有効期限を設定する必要はありません。つまり、Expiration Date プロパティの値は 0 になります。
   - **[パスワード]** – 上で入力したユーザ・アカウントに関連付けられたパスワード。
   - **[プロダクト]** – InterSystems IRIS。
   - **[サービスプリンシパル名]** – これには値を指定しないでください(このフィールドは、Kerberos で使用するためにゲートウェイを構成するときに使用します)。
   - **[キーテーブル]** – これには値を指定しないでください(このフィールドは、Kerberos で使用するためにゲートウェイを構成するときに使用します)。

   これらの値をすべて入力した後、**[設定を保存]** ボタンをクリックすると、入力した値が保存されます。

ゲートウェイに対する認証要件と、そのゲートウェイを使用するアプリケーションに対する認証要件との間に直接の関係はない点に注意する必要があります。例えば、Kerberos 認証を使用するようにゲートウェイを構成していても、Web アプリケーションに対する認証メカニズムとしてインスタンス認証を要求できます。ゲートウェイに対して認証方法をまったく構成していなくても同様です。実際、ゲートウェイそのものに対して特定の認証メカニズムを選択しても、Web アプリケーションに対して技術的な要件は発生しません。逆に Web アプリケーションに対する認証メカニズムを選択しても、ゲートウェイに対する要件は発生しません。同時に、CSP では他の場合に比べ、ペア化が発生する可能性が高くなります。Web アプリケーションで Kerberos 認証を使用している場合、ゲートウェイに対して他の形式の認証を使用すると、暗号化されていないチャンネルを通じて Kerberos 認証の情報が流れることになります。その結果、Kerberos 認証の効果が低下する可能性があります。

インスタンス認証を使用している Web アプリケーションでは、エンドユーザのユーザ名とパスワードはブラウザから Web サーバに渡され、さらに Web ゲートウェイに渡されます。このゲートウェイは InterSystems IRIS サーバとの独自の接続を備えているので、ユーザ名とパスワードはその InterSystems IRIS サーバに渡されます。ゲートウェイと InterSystems IRIS サーバとの接続を確立するために、このゲートウェイでは CSPSystem アカウントが使用されます。このアカウントは、InterSystems IRIS の事前定義アカウントの 1 つです。

既定では、これらのトランザクションはどれも暗号化されていません。TLS を使用すれば、ブラウザから Web サーバに送信されるメッセージを暗号化できます。Kerberos を使用すると、"Web 接続で使用するセキュア・チャンネルの設定" で説明したように、ゲートウェイから InterSystems IRIS サーバに送信されるメッセージを暗号化できます。Kerberos を使用していない場合は、ゲートウェイのホスト・マシンと InterSystems IRIS サーバのホスト・マシンとの間の接続を物理的に保護することをお勧めします。例えば、両方のマシンをロックされた同じ領域に置いて、両者間で直接的な物理接続を確立することにより、接続を保護できます。

アーキテクチャに関係なく、各接続でデータが安全であることを確認してください。

ODBC

InterSystems IRIS では、サポート対象のすべてのプラットフォーム間で ODBC 接続に対するインスタンス認証がサポートされています。この場合は、クライアント側の構成が必要です。クライアントの動作を構成する手順は、以下のようにプラットフォームによって異なります。

- Windows 以外のプラットフォームでは、InterSystems ODBC 初期化ファイルを使用して、接続情報を提供する名前と値の対を指定します。このファイルについては、"InterSystems ODBC ドライバの使用法" で概要を説明しています。このファイルには、インスタンス認証に関連する以下の変数が含まれています。
  - Authentication Method ― ODBC クライアントを DSN に認証する方法を指定します。0 はインスタンス認証、1 は Kerberos を指定します。
  - UID ― DSN への接続に使用する既定のユーザ・アカウントの名前を指定します。実行時には、アプリケーションの動作に応じて、エンドユーザはこの値を別のユーザ・アカウントに置き換えることができます。
  - Password ― 既定のユーザ・アカウントに関連付けられたパスワードを指定します。エンドユーザによる UID 値の変更が許可されている場合は、新しく指定されたユーザのパスワードがアプリケーションで受け入れられます。
- Windows クライアントに対する接続情報は、GUI を使用した指定、またはプログラムによる指定のいずれかが可能です。
  - GUI では、ODBC DSN を構成するダイアログを使用します。**[システム DSN]** タブにオプションが表示されます。この画面には、それぞれのフィールドを説明するヘルプがあります。Windows の [スタート] メニューからの操作でこの画面を表示する方法は、Windows のバージョンによって異なりますが、多くの場合は **[コントロール パネル]** の **[管理ツール]** で **[データ・ソース (ODBC)]** の画面に表示されます。
  - プログラムで指定する場合は、名前と値の組み合わせのセットを受け取る SQLDriverConnect 関数を使用できます。SQLDriverConnect は、ODBC API の一部である C 呼び出しです。この名前と値の組み合わせは、パスワードが PWD キーワードで識別される点を除けば、Windows 以外のプラットフォームで使用できる初期化ファイルにあるものと同じです。

Telnet

クライアントで Windows 用の InterSystems IRIS Telnet を使用して接続を確立するには、InterSystems IRIS リモート・サーバの一部として格納されている構成情報を使用します。リモート・サーバを構成するには、クライアント・マシンに移動します。そのマシンで以下の手順を実行します。

1. InterSystems IRIS ランチャーをクリックし、メニューから **[優先接続サーバ]** を選択します (**[優先接続サーバ]** の選択対象には、現在の優先接続サーバの名前も表示されています)。
2. 表示されたサブメニューから、**[追加/編集]** を選択します。
3. リモート・サーバを新規に作成するには、**[追加]** ボタンをクリックします。既存のサーバを構成するには、接続先の InterSystems IRIS サーバを選択し、**[編集]** ボタンをクリックします。
4. **[接続を追加]** ダイアログが表示されます。このダイアログの **[認証方法]** 領域で、インスタンス認証を指定する **[パスワード]** をクリックします。
5. 既存のサーバの値を編集する場合は、このダイアログにある一般的な内容のフィールドで値を変更したり、追加したりする必要はありません。これらの値は、編集するサーバを選択することで自動的に決まります。

   新規サーバを追加する場合に入力するフィールドの詳細は、"リモート・サーバ接続の定義" を参照してください。
6. [OK] をクリックすると、指定した値が保存され、ダイアログが閉じます。

重要　Windows 以外で稼動しているマシンに Telnet を使用して接続しても InterSystems IRIS Telnet サーバは使用できません。この場合は、オペレーティング・システム付属の Telnet サーバを使用します。サーバ・マシンとの接続が確立されれば、**%Service_Terminal** サービスを使用して InterSystems IRIS に接続できます。

代行認証

代行認証の背景

InterSystems IRIS では、代行認証がサポートされます。代行認証を使用すると、企業の既存の認証システムなど、カスタムのメカニズムを実装して、インターシステムズのセキュリティに含まれる認証アクティビティやロール管理アクティビティを置き換えることができます。アプリケーション開発者は、代行認証コードのコンテンツを完全に制御します。代行認証は、InterSystems IRIS のインスタンスの %SYS ネームスペースで ZAUTHENTICATE ルーチンが検出された場合に行われます。このようなルーチンが存在する場合、InterSystems IRIS はこれを使用して、新規コードと既存のコードのいずれかを呼び出してユーザを認証します。InterSystems IRIS には **ZAUTHENTICATE.mac** ルーチンがあり、これは ZAUTHENTICATE ルーチンを作成するためのテンプレートとして機能します。

重要　　HealthShare® で認証を使用している場合は、インターシステムズが提供する ZAUTHENTICATE ルーチンを使用する必要があります。独自のルーチンは作成できません。

代行認証の動作

ユーザがログインを試行して、InterSystems IRIS が代行認証を呼び出す場合、イベントのシーケンスは以下のようになります。

1. サービスまたはアプリケーションが代行認証を使用する場合、ログイン試行によって ZAUTHENTICATE ルーチンが自動的に呼び出されます。このルーチンの認証コードは、任意のユーザ定義の ObjectScript、クラス・メソッド、または $ZF コールアウト・コードにできます。
2. 次に行う手順は、認証が成功するかどうか、および今回が ZAUTHENTICATE を使用した最初のログインであるかどうかによって異なります。
   - ZAUTHENTICATE が成功し、今回がこのメカニズムを使用した初めてのユーザ認証である場合、そのユーザはタイプを“代行ユーザ”として InterSystems IRIS ユーザのリストに追加されます。ZAUTHENTICATE によってロールまたは他の特性が設定されると、そのロールまたは特性がそのユーザのプロパティの一部になります。
   - ZAUTHENTICATE が成功し、今回が最初のログインでない場合、ZAUTHENTICATE によってそのユーザのプロパティが更新されます。
   - ZAUTHENTICATE が失敗すると、アクセス拒否のエラーが表示されます。ユーザはシステムにアクセスできません。この原因を調べる手順は次のとおりです。
     a. ユーザ・プロファイルの **[ログイン失敗の理由]** フィールドの内容をチェックします。
     b. さらに、監査ログで %System/%Login/LoginFailure イベントをチェックして、詳細な情報を確認します。監査や LoginFailure イベントが有効になっていない場合は、これらを両方とも有効にしてログイン失敗の状況を再現する必要がある場合があります。
3. インスタンスと関連サービスの 2 要素認証が有効な場合は、ユーザの **PhoneNumber** および **PhoneProvider** プロパティが設定されていることが確認されます。これらのプロパティが設定されている場合は、2 要素認証が実行されます。これらのプロパティが設定されていない場合は、2 要素認証に進むことはできず、ユーザは認証されません。
4. **[ユーザ]** ページ (**[システム管理]** > **[セキュリティ]** > **[ユーザ]**) で、ユーザ・リストの [タイプ] 列に代行ユーザとしてユーザが表示されます。管理ポータルに表示されるユーザのプロパティは読み取り専用です。InterSystems IRIS から編集することはできません (これらの情報はすべて InterSystems IRIS の外部から取得されるため)。

   注釈　　InterSystems IRIS パスワード・ユーザが、同時に代行ユーザになることはできません。

代行認証の構成の概要

代行認証を使用するには、以下の手順を実行します。

1. ZAUTHENTICATE ルーチンで、ユーザ定義の認証コードを作成します。これには、2 要素認証を使用できます。このルーチンは、ロールや他のユーザ・プロパティの指定など、ユーザ・アカウントの基本的な設定も実行することができます。

   HealthShare Health Connect を使用している場合は、このドキュメントの説明に従って ZAUTHENTICATE カスタム・ルーチンを作成します。

   HealthShare Unified Care Record を使用している場合、Unified Care Record には独自のバージョンのルーチンが付属しているため、カスタム・バージョンの ZAUTHENTICATE を作成して代行認証を実装することはできません。代わりに、クラス HS.Local.ZAUTHENTICATE のメソッドをカスタマイズする必要があります。詳細は、ブック "Authenticating Users in Unified Care Record" の "Customizing Authentication via Local ZAUTHENTICATE" を参照してください。
2. [認証オプション] ページで、InterSystems IRIS インスタンスに対して代行認証を有効にします。
3. 必要に応じて、関連するサービス、アプリケーション、またはその両方に対して代行認証を有効にします。
4. オプションとして、InterSystems IRIS インスタンスおよび必要に応じて Web アプリケーションとクライアント・サーバ・アプリケーションで、2 要素認証を有効化します。

例えば、インスタンスの管理ポータルで代行認証を使用するには、以下の手順を実行します。

1. ZAUTHENTICATE で、ユーザ定義の認証コードを作成します。
2. InterSystems IRIS インスタンス全体の代行認証を有効にします。
3. /csp/sys* アプリケーションのセットの代行認証を有効にします。

代行 (ユーザ定義) 認証コードの作成

ここでは、ユーザ専用の ZAUTHENTICATE ルーチン作成のさまざまな側面について説明します。

- 認証コードの基礎
- シグニチャ
- 認証コード
- ロールと他のユーザ特性の値の設定
- 返り値とエラー・メッセージ

認証コードの基礎

インターシステムズが提供するサンプル・ルーチン ZAUTHENTICATE.mac をコピーして変更することができます。このルーチンは GitHub の Samples-Security サンプルに含まれています (https://github.com/intersystems/Samples-Security)。"InterSystems IRIS で使用するサンプルのダウンロード" で説明されているようにサンプル全体をダウンロードすることもできますが、単に GitHub でルーチンを開いて、その内容をコピーする方が簡単です。

独自の ZAUTHENTICATE.mac を作成するには、以下の手順を実行します。

1. ZAUTHENTICATE.mac をテンプレートとして使用するには、その内容を %SYS ネームスペースの ZAUTHENTICATE.mac ルーチンにコピーして保存します。
2. ZAUTHENTICATE.mac サンプル内のコメントを確認します。そこには、カスタム版のルーチンを実装する方法に関する重要なガイダンスが記載されています。
3. ユーザ・アカウントの特性を設定するには、カスタム認証コードと必要なコードを追加してルーチンを編集します。

注意　InterSystems IRIS では ZAUTHENTICATE の認証コードに対する制約を行わないため、アプリケーション・プログラマは、コードが十分に安全であるかどうかを確認する必要があります。

シグニチャ

ZAUTHENTICATE のシグニチャは以下のとおりです。

ObjectScript

```
ZAUTHENTICATE(ServiceName, Namespace, Username, Password, Credentials,
        Properties) PUBLIC {

    // authentication code
    // optional code to specify user account properties and roles
}
```

以下はその説明です。

- ServiceName ー ユーザから InterSystems IRIS への接続を仲介しているサービスの名前を表す文字列 (**%Service_Console** や **%Service_WebGateway** など)。
- Namespace ー 目的とする接続先の InterSystems IRIS サーバにあるネームスペースを表す文字列。これは、スタジオや ODBC など、**%Service_Bindings** サービスで使用します。
- Username ー ルーチンのコードによって検証されるユーザが入力したアカウントの名前を表す文字列。
- Password ー 検証されるユーザが入力したパスワードを表す文字列。
- Credentials ー 参照渡し。今回のバージョンの InterSystems IRIS では、実装されていません。
- Properties ー 参照渡し。Username で指定したアカウントの特性を定義する返り値の配列。

InterSystems IRIS が ZAUTHENTICATE を呼び出すと、これらの引数の値がルーチンに提供されます。

注釈　インターシステムズ製品の古いバージョンでは、ZAUTHENTICATE は引数を 4 つ取ります。下位互換性を保つために、引き続き引数が 4 つのバージョンを使用できます。コードを古いバージョンから新しいバージョンに更新する場合、新しい引数は 2 番目と 5 番目になることに注意してください。すなわち、Namespace と Credentials です。

認証コード

認証コードのコンテンツはアプリケーションに固有です。認証が成功すると、ルーチンから $$$OK マクロが返されます。失敗した場合はエラー・コードが返されます。返り値の詳細は、"返り値とエラー・メッセージ" を参照してください。

注意　InterSystems IRIS では ZAUTHENTICATE の認証コードに対する制約を行わないため、アプリケーション・プログラマは、コードが十分に安全であるかどうかを確認する必要があります。

GetCredentials エントリ・ポイント

ZAUTHENTICATE は GetCredentials エントリ・ポイントを含んでいます。エントリ・ポイントは代行認証がサービスに対して有効となるたびに、ユーザ名およびパスワードの入力が要求される前に呼び出されます。ユーザからユーザ名およびパスワードを得る代わりに、(アプリケーション開発者が作成した) 関数のコードがユーザ名およびパスワードを指定します。その後、返されたユーザ名およびパスワードは、ユーザが入力したかのように通常の方法にて認証されます。このメカニズムを使用することで、ユーザ名およびパスワードをエントリ・ポイント内さらには認証コード内で、プロセスの $roles に対して提供することができます。

このエントリ・ポイントより返されるユーザ名とパスワードは、アプリケーション開発者が選択する任意のメカニズムにより取得できます。これらは、グローバルから取得することも、外部の DLL または LDAP 呼び出しから取得することも、または単純にルーチン内で設定することもできます。さらに、アプリケーション開発者は、ターミナル接続やログイン・ページなどでユーザ名およびパスワードの入力を要求するコードを提供することもできます。

GetCredentials エントリ・ポイントの呼び出しがある場合、戻り値およびその他の要素によって次の動作が決まります。

- コードが Username および Password の値を設定し、成功ステータス (**$$$OK**) も返した場合は、以下のように動作します。
  - さらにユーザ名とパスワードの入力が求められることはありません。
  - 認証プロセスが続行されます。

  重要　アクセス・ポイントから **$$$OK** が返された場合、コードは Username および Password の値を設定する必要があります。それ以外の場合、ユーザはシステムへのアクセスを拒否され、エラーが監査ログに書き込まれます。

- エントリ・ポイントからエラー・ステータス **$SYSTEM.Status.Error($$$GetCredentialsFailed)** が返された場合は、通常のユーザ名とパスワードの入力が求められます。
- エントリ・ポイントからその他のエラー・ステータスが返された場合は、以下のように動作します。
  - ユーザはシステムへのアクセスを拒否されます。
  - エラーが監査ログに記録されます。

GetCredentials エントリ・ポイントの以下の例では、コードは各種サービスに対してさまざまな動作を実行します。

- **%Service_Console** では、ユーザに情報を要求せずに、プロセスのユーザ名とパスワードをそれぞれ **_SYSTEM** および **SYS** に設定します。
- **%Service_Bindings** では、ユーザにユーザ名とパスワードの入力を強制します。
- Web アプリケーションでは、使用中のアプリケーションが **/csp/** サンプル・アプリケーションであるかどうかをチェックします。そうである場合は、ユーザ名とパスワードを **AdminUser** および **Test** に設定します。他の Web アプリケーションである場合は、アクセスはすべて拒否されます。
- その他のどのサービスの場合も、アクセスは拒否されます。

最後に、Error エントリ・ポイントは必要に応じてクリーンアップを行います。

このコードは以下のとおりです。

ObjectScript

```
GetCredentials(ServiceName,Namespace,Username,Password,Credentials) Public {

  // For console sessions, authenticate as _SYSTEM.
  If ServiceName="%Service_Console" {
    Set Username="_SYSTEM"
    Set Password="SYS"
    Quit $SYSTEM.Status.OK()
  }

  // For a web application, authenticate as AdminUser.
  If $isobject($get(%request)) {
    If %request.Application="/csp/samples/" {
      Set Username="AdminUser"
      Set Password="Test"
      Quit $System.Status.OK()
    }
  }

  // For bindings connections, use regular prompting.
  If ServiceName="%Service_Bindings" {
    Quit $SYSTEM.Status.Error($$$GetCredentialsFailed)
  }

  // For all other connections, deny access.
  Quit $SYSTEM.Status.Error($$$AccessDenied)
}
```

詳細は、ZAUTHENTICATE.mac の、このエントリ・ポイントのコメントを参照してください。

SendTwoFactorToken エントリ・ポイント

ZAUTHENTICATE は SendTwoFactorToken エントリ・ポイントを含んでいます。このエントリ・ポイントは 2 要素認証と共に使用します。これが定義されており、InterSystems IRIS インスタンスで 2 要素認証が有効な場合、インスタンスがユーザの携帯電話に送信するメッセージおよびトークンの形式の既定のシステム設定をオーバーライドできます。これによって、同じ InterSystems IRIS インスタンス上でもアプリケーションごとに異なるメッセージを使用できます。

このエントリ・ポイントの詳細と使用例は、サンプル ZAUTHENTICATE.mac でこのエントリ・ポイントを参照してください。

ロールと他のユーザ特性の値の設定

最初の認証が成功すると、ZAUTHENTICATE は認証されたユーザのロールと他の特性を確立できます。以降のログインでは、ユーザ・レコードのこれらの要素を更新できます。

これを行うために、ZAUTHENTICATE のコードにより Properties 配列の値が設定されます(Properties は、ZAUTHENTICATE への参照によって渡されます)。通常、設定する値のソースは、ZAUTHENTICATE が使用できるユーザ情報のリポジトリです。

ユーザのプロパティ

Properties 配列の要素は以下のとおりです。

- Properties("Comment") ー 任意のテキスト
- Properties("FullName") ー ユーザの名前と姓
- Properties("NameSpace") ー ターミナル・ログインの既定のネームスペース
- Properties("Roles") ー ユーザが InterSystems IRIS で保持するコンマ区切りのロール・リスト。
- Properties("Routine") ー ターミナル・ログインに対して実行されるルーチン
- Properties("Password") ー ユーザのパスワード
- Properties("Username") ー ユーザのユーザ名
- Properties("PhoneNumber") ー ユーザの携帯電話番号で、2 要素認証で使用
- Properties("PhoneProvider") ー ユーザの携帯電話のサービス・プロバイダで、2 要素認証で使用

各要素については、この後のセクションでそれぞれ詳しく説明します。

注釈　プロパティの配列の各要素の値によって、それぞれの要素に関連する、認証対象のユーザのプロパティ値が設定されます。これらのプロパティのサブセットのみを使用したり、認証後にプロパティ値を操作したりすることはできません。

Comment

ZAUTHENTICATE で Properties("Comment") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Comment プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Comment の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

FullName

ZAUTHENTICATE で Properties("FullName") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Full name プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Full name の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

Namespace

ZAUTHENTICATE で Properties("Namespace") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Startup Namespace プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Startup Namespace の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

InterSystems IRIS に接続すると、Startup Namespace の値 (Properties("Namespace") の値) は、ローカル・アクセス (コンソール、ターミナル、Telnet など) に認証されたユーザの最初のネームスペースを決定します。Startup Namespace に値が指定されない場合 (Properties("Namespace") に値が指定されないため)、ローカル・アクセスに認証されたユーザの最初のネームスペースは以下のように決定されます。

1. USER ネームスペースが存在する場合、これが最初のネームスペースになります。
2. USER ネームスペースが存在しない場合、最初のネームスペースは %SYS ネームスペースになります。

注釈　ユーザが最初のネームスペースに対する適切な特権を保持していない場合、アクセスは拒否されます。

Password

ZAUTHENTICATE で Properties("Password") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Password プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Password の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

Roles

ZAUTHENTICATE で Properties("Roles") の値を設定すると、その文字列によってユーザの割り当て先の Roles が指定されます。この値はコンマ区切りのロール・リストを含む文字列です。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Roles の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。ユーザのロールに関する情報は、**[ユーザ編集]** ページの **[ロール]** タブで確認できます。

Properties("Roles") で返されるロールが定義されない場合、ユーザはロールに割り当てられません。

したがって、ログインしたユーザは以下のようにロールに割り当てられます。

- ロールが Properties("Roles") に示され、InterSystems IRIS インスタンスで定義される場合、ユーザはそのロールに割り当てられます。
- ロールが Properties("Roles") に示され、InterSystems IRIS インスタンスで定義されない場合、ユーザはそのロールに割り当てられません。
- ユーザは、_PUBLIC ユーザに関連付けられたロールには常に割り当てられます。また、すべてのパブリック・リソースにアクセスできます。_PUBLIC ユーザの詳細は、"_PUBLIC アカウント" を参照してください。パブリック・リソースの詳細は、"サービスとそのリソース" を参照してください。

Routine

ZAUTHENTICATE で Properties("Routine") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Startup Tag^Routine プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Startup Tag^Routine の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

Properties("Routine") に値がある場合、この値によって、ターミナル・タイプのサービス (コンソール、ターミナル、Telnet など) でログイン後に自動的に実行するルーチンが指定されます。Properties("Routine") に値を指定しない場合、ログインはプログラマ・モードでターミナル・セッションを開始します。

Username

ZAUTHENTICATE で Username プロパティを返す場合、Username の値は関数での処理の後にセキュリティ・データベースに書き込まれます。このため、ユーザがプロンプトで入力した値が変更される可能性があります。ZAUTHENTICATE で Username プロパティを返さない場合、プロパティの値は入力時にセキュリティ・データベースに書き込まれます。

ZAUTHENTICATE で Properties("Username") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Name プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。これにより、アプリケーション・プログラマは、ログイン・プロンプトでエンドユーザが入力した内容を正規化できます。

Properties("Username") の値を呼び出し元のルーチンに渡す明示的な呼び出しがない場合、正規化は行われず、プロンプトでエンドユーザが入力する値が、変更されずに、そのユーザ・アカウントの Name プロパティの値としてそのまま使用されます。

PhoneNumber と PhoneProvider

これらは 2 要素認証と関連付けられたプロパティです。

ZAUTHENTICATE が Properties("PhoneNumber") と Properties("PhoneProvider") の値を設定すると、これらはユーザの携帯電話番号および携帯電話サービス・プロバイダとしてユーザの InterSystems IRIS データベースに書き込まれます。これらが呼び出しルーチンに渡されない場合、InterSystems IRIS データベースに書き込まれた電話番号とサービス・プロバイダは NULL 文字列です。したがって、代行認証で 2 要素認証を使用するには、これらの両方を提供する必要があります。

ユーザ情報のリポジトリ

ZAUTHENTICATE は、グローバルや外部ファイルなど、ユーザ情報のリポジトリであればどのような種類でも参照できます。認証されたユーザをこの情報で作成または更新できるように、ルーチンのコードによって Properties 配列で外部プロパティを設定します。例えば、あるリポジトリにロールやネームスペースなどの情報が含まれる一方で、ZAUTHENTICATE コードは InterSystems IRIS がその情報を利用できるようにする必要があります。

リポジトリ内の情報が変更されると、このアクションを実行するコードが ZAUTHENTICATE にある場合、この情報は InterSystems IRIS ユーザ情報に伝播されます。また、このようなコードがある場合、リポジトリでユーザのロールが変更されなければなりません。セッション中にユーザのロールを変更した場合、その変更は次のログインまで有効になりません。次のログインの時点で、そのユーザのロールは ZAUTHENTICATE によってリセットされます。

返り値とエラー・メッセージ

ルーチンは以下のいずれかの値を返します。

- 成功 – $$$OK。ユーザ名とパスワードの組み合わせが正常に認証されたことを示します。
- 失敗 – $SYSTEM.Status.Error($$$ERRORMESSAGE)。認証が失敗したことを示します。

ZAUTHENTICATE は、システム定義またはアプリケーション固有のエラー・メッセージを返すことができます。これらのメッセージはすべて、**%SYSTEM.Status** クラスの Error メソッドを使用します。このメソッドは、$SYSTEM.Status.Error として呼び出され、エラーの状況に応じて、1 つまたは 2 つの引数を取ります。

使用可能なシステム定義のエラー・メッセージは以下のとおりです。

- $SYSTEM.Status.Error($$$AccessDenied) – “アクセスが拒否されました” のエラー・メッセージ
- $SYSTEM.Status.Error($$$InvalidUsernameOrPassword) – “ユーザ名またはパスワードが無効です” のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserNotAuthorizedOnSystem,Username) – “ユーザ Username は許可されていません” のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserAccountIsDisabled,Username) – “ユーザ Username アカウントが無効です” のエラー・メッセージ

- $SYSTEM.Status.Error($$$UserInvalidUsernameOrPassword,Username) ー“ユーザ Username の名前またはパスワードは無効です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserLoginTimeout) ー“ログインタイムアウト”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserCTRLC) ー“ログインは中止されました”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserDoesNotExist,Username) ー“ユーザ Username は存在しません”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserInvalid,Username) ー“ユーザ名 Username が無効です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$PasswordChangeRequired) ー“パスワードの変更が必要です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserAccountIsExpired,Username) ー“ユーザ Username のアカウントは失効しました”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserAccountIsInactive,Username) ー“ユーザ Username のアカウントはアクティブではありません”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserInvalidPassword) ー“無効なパスワードです”のエラー・メッセージ
- $SYSTEM.Status.Error($$$ServiceDisabled,ServiceName) ー“サービス Servicename のログインは無効です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$ServiceLoginsDisabled) ー“ログインは無効です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$ServiceNotAuthorized,ServiceName) ー“ユーザはサービスを許可されていません”のエラー・メッセージ

カスタム・メッセージを生成するには、$SYSTEM.Status.Error() メソッドを使用して、このメソッドに `$$$GeneralError` マクロを渡し、2 番目の引数として任意のカスタム・テキストを指定します。以下に例を示します。

```
$SYSTEM.Status.Error($$$GeneralError,"Any text here")
```

エラー・メッセージが呼び出し元に返されると、そのメッセージは監査データベース (LoginFailure イベント監査が有効な場合) にログとして記録されます。表示されるエラー・メッセージは $SYSTEM.Status.Error($$$AccessDenied) のみです。ただし、`$$$PasswordChangeRequired` エラーのメッセージも表示されます。現在のパスワードから新しいパスワードへの変更をユーザに要求する場合、このエラーを返します。

代行認証の設定

認証 (および、オプションで承認タスク) を実行する ZAUTHENTICATE ルーチンを作成したら、次に、インスタンスの関連サービスまたはアプリケーションでそのルーチンを有効にします。手順は以下のとおりです。

1. インスタンス全体の代行認証を有効にします。**[認証/Web セッション・オプション]** ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[認証/Web セッション・オプション]**) で、**[代行認証を許可]** を選択し、**[保存]** をクリックします。

   インスタンスで代行認証を有効にすると、関連サービスの **[サービス編集]** ページと、関連アプリケーションの **[ウェブ・アプリケーション編集]** ページに **[代行]** チェック・ボックスが表示されます。
2. 必要に応じて、サービスとアプリケーションの代行認証を有効にします。

以下のサービスは認証代行をサポートしています。

- **%Service_Bindings**
- **%Service_CallIn**
- **%Service_ComPort**
- **%Service_Console**

- **%Service_Login**
- **%Service_Terminal**
- **%Service_Telnet**
- **%Service_WebGateway**

これらのサービスは、アクセス・モードによっていくつかのカテゴリに分類されます。

- ローカル・アクセス ー

  %Service_CallIn、**%Service_ComPort**、**%Service_Console**、**%Service_Login**、**%Service_Terminal**、**%Service_Telnet**

  ローカル接続で代行認証を使用するには、サービスの代行認証を有効にします。

- クライアント・サーバ・アクセス ー

  %Service_Bindings

  クライアント・サーバ接続で代行認証を使用するには、サービスの代行認証を有効にします。

- Web アクセス ー

  %Service_WebGateway

  Web ベースの接続で代行認証を使用するには、Web アプリケーションの代行認証を有効にします。また、サービス **%Service_WebGateway** を有効にすることで、Web ゲートウェイでこれを有効にすることもできます。

代行認証成功後の注意事項

ユーザが認証された後の重要な項目は以下のとおりです。

- システムの状態
- パスワードの変更

システムの状態

代行認証を使用して最初に認証されるユーザは、“代行ユーザ” というタイプで **[ユーザ]** ページ (**[システム管理]** > **[セキュリティ]** > **[ユーザ]**) のユーザ・テーブルに表示されます。システム管理者が管理ポータル (またはその他の InterSystems IRIS ネイティブ機能) を使用して明示的にユーザを作成した場合、そのユーザのタイプは “InterSystems IRIS パスワード・ユーザ” になります。ユーザが代行認証を使用してログインを試行し、認証が正常に行われる場合でも、そのユーザが既に (代行ユーザではなく) InterSystems IRIS ユーザとして存在することを InterSystems IRIS が検出すると、ログインは失敗します。

注釈　シャード・クラスタでシャード操作を実行するには、代行ユーザがシャード・クラスタの各ノードで内部シャード接続以外の方法によって事前に認証されている必要があります。シャードの詳細は、″シャーディングによるデータ量に応じた InterSystems IRIS の水平方向の拡張″ を参照してください。

パスワードの変更

ZAUTHENTICATE ルーチンには、エントリ・ポイント ChangePassword も含まれており、そこにはユーザのパスワードを変更するコードが含まれています。このエントリ・ポイントのシグニチャは以下のとおりです。

ObjectScript

```
ChangePassword(Username,NewPassword,OldPassword,Status) Public {}
```

以下はその説明です。

- Username は、パスワードを変更するユーザを指定する文字列です。
- NewPassword は、ユーザのパスワードの新しい値を指定する文字列です。
- OldPassword は、ユーザのパスワードの古い値を指定する文字列です。
- Status (参照によって渡される) は、InterSystems IRIS の状態値を受信します。受信した値では、パスワード変更が成功したことを示すか、ルーチンの失敗の原因となったエラーを指定します。

代行認証または他のメカニズムでの LDAP の使用

カスタム認証システムの一部として (つまり、InterSystems IRIS の代行認証機能で) LDAP を使用することもできます。そのためには、ZAUTHENTICATE ルーチンのカスタム認証コードの一部として **%SYS.LDAP** クラスの呼び出しを使用します。

インターシステムズでは、これらの呼び出し方法を示すサンプル・ルーチン **LDAP.mac** を提供しています。このルーチンは GitHub の Samples-Security サンプルに含まれています (https://github.com/intersystems/Samples-Security)。

さらに、LDAP に対して認証する必要がある場合、または別のメカニズムによって認証情報を収集した後にインスタンス認証を使用する必要がある場合は、それらの認証情報で $SYSTEM.Security.Login を呼び出してユーザを認証します。

2 要素認証

2 要素認証

使用中の認証メカニズムの他に、InterSystems IRIS は 2 要素認証の使用をサポートしています。つまり、インターシステムズの認証では、エンドユーザが 2 つの別々のエレメントつまり“要素”を持つことを要求できます。エンドユーザの観点では、最初の要素はユーザが知っているもの (パスワードなど)、2 番目の要素はユーザが持っているもの (スマートフォンなど) です。InterSystems IRIS は、以下の 2 つのメカニズムのいずれかを使用してエンドユーザの 2 要素認証を実行します。

- SMS テキスト認証 – InterSystems IRIS はセキュリティ・コードをエンドユーザの電話に SMS で送ります。エンドユーザは、指示に従ってそのコードを入力します。
- タイムベース・ワンタイム・パスワード (TOTP) – エンドユーザは最初に InterSystems IRIS から秘密鍵を受け取ります。この鍵は、InterSystems IRIS とエンドユーザのアプリケーション (携帯電話のアプリケーションなど) または物理的な認証デバイスとの間の共有秘密です。両者が、この鍵およびその他の情報を使用して、検証コードとして作用し、InterSystems IRIS の指示に従ってエンドユーザが入力する TOTP を生成します。TOTP は 60 秒後に期限切れになり、エンドユーザはこれを 1 回しか使用できません。これがタイムベースおよびワンタイムと呼ばれている理由です。

このセクションでは、以下のトピックについて説明します。

- 2 要素認証の設定の概要
- サーバの 2 要素認証の構成
- サービスに対する 2 要素認証の有効化または無効化
- 2 要素認証向けの Web アプリケーションの構成
- 2 要素認証向けのエンドユーザの構成
- 2 要素認証向けのバインディング・クライアントの構成

2 要素認証の設定の概要

2 要素認証の設定の主な手順は以下のとおりです。

1. インスタンス全体に対する 2 要素認証の有効化および構成を行います。インスタンスを構成して SMS テキスト認証、TOTP 認証、またはその両方を使用できます。TOTP 認証の詳細は、"2 要素の TOTP の概要" を参照してください。
2. SMS テキスト認証の場合、必要に応じて携帯電話サービス・プロバイダの構成を行います。これには以下が含まれます。
   - ある携帯電話サービス・プロバイダが必要であるのに、既定のプロバイダのリストに含まれていない場合、このサービス・プロバイダを追加する。
   - 既存のプロバイダに対し、必要に応じて構成情報を変更する (既定または追加)。
3. 以下のサービスを適切に構成します。
   - **%Service_Bindings** – サービスに対して 2 要素認証を有効化し、次の手順に進みます。
   - **%Service_Console** および **%Service_Terminal** – 単にサービスに対して 2 要素認証を有効化します。必要な手順はこれですべてです。
   - **%Service_WebGateway** – **%Service_WebGateway** の 2 要素認証を一元的に有効にする方法はありません。次の手順に進みます。

各サービスに対していずれかまたは両方の認証のタイプを有効にできます。サービスの詳細は、"サービス"を参照してください。

4. クライアント・サーバ・アプリケーションおよび Web アプリケーションを適切に構成します。

   a. クライアント・サーバ・アプリケーション (**%Service_Bindings** を使用するアプリケーション) の場合、2 要素認証をサポートするクライアント・アプリケーションに適切な呼び出しを追加します。これは、使用中のクライアント側のコンポーネント (Java、JDBC、.NET など) に応じて異なるプログラミング・タスクです。

      重要　2 要素認証は、人のエンドユーザからの応答をリアルタイムで受け取るように設計されています。1 つのセッションが実際には複数の連続セッションから構成されているとエンドユーザが見なしている場合、2 番目の要素を繰り返し要求することは、予期しない困難なユーザ・エクスペリエンスをもたらすことになる場合があります。クライアント・サーバ・アプリケーションでは、基礎プロトコルによって、クライアントが接続の確立、切断、再確立を繰り返し強いられることがよくあります。この種のアプリケーションでは、このような作業のために、2 要素認証の使用は不適切になります。

   b. Web アプリケーション (**%Service_WebGateway** を使用するアプリケーション) については、2 要素認証をサポートするように各アプリケーションを構成します。

   注釈　Windows では **%Service_Console** サービスを使用し、他のオペレーティング・システムでは **%Service_Terminal** サービスを使用する InterSystems IRIS ターミナルの場合、サーバ側の設定の他に必要な構成はありません。InterSystems IRIS は、これらのシステムにおいてプロンプトを制御するため、2 要素認証プロンプトを使用して (認証メカニズムに関係なく) 標準プロンプトに従い、エンドユーザ入力をこれに応じて処理するだけです。

5. 代行認証を使用する場合は、必要に応じて ZAUTHENTICATE.mac ルーチンを変更します。詳細は "代行認証の使用法" を参照してください。

6. 各エンドユーザを構成し、SMS テキスト認証または TOTP 認証を有効にします。エンドユーザは、両方のメカニズムを使用するように構成できますが、同時に両方のメカニズムを有効にすることはできません。

2 要素の TOTP の概要

タイムベース・ワンタイム・パスワード (TOTP) 認証を使用した 2 要素認証は以下のように動作します。

1. TOTP を生成する認証デバイスまたはアプリケーションを選択し、そのデバイスまたはアプリケーションを提供するか、ユーザがそのデバイスまたはアプリケーションを所持していることを確認します。

2. 2 要素の TOTP 認証向けにエンドユーザを構成すると、システムは秘密鍵を生成します。この鍵は、Base 32 でエンコードされてランダム化されたビット文字列として表示されます。InterSystems IRIS とエンドユーザはこの秘密鍵を共有します (これが共有秘密と呼ばれる理由です)。InterSystems IRIS とエンドユーザの両方の認証デバイスまたはアプリケーションはこの秘密鍵を使用して、検証コードとして作用する TOTP 自体を生成します。**[検証コード]** フィールドまたはプロンプトにエンドユーザが入力する TOTP は 6 桁の文字列で、定期的 (既定では 30 秒毎) に新しいものが生成されます。

3. ログイン時、エンドユーザが InterSystems IRIS にパスワードを入力した後、InterSystems IRIS は追加で TOTP の入力を求めます。エンドユーザが TOTP を入力すると、ログイン・プロセスが完了します。

エンドユーザは、以下のいくつかの方法で InterSystems IRIS から秘密鍵を取得できます。

- 2 要素の TOTP 認証をサポートするようにエンドユーザのアカウントを構成するとき、エンドユーザの **[ユーザ編集]** ページにエンドユーザの秘密鍵が発行者の名前およびエンドユーザのアカウント名と共に表示されます。上記の情報をすべて含む QR コードも表示されます (QR コードは下の写真のようなマシンが読むことができるコードです)。ここでエンドユーザは、コードをスキャンするか情報を手入力することによって、情報を認証デバイスまたはアプリケーションに入力できます。

- Web アプリケーションまたはターミナル・セッションにログインするとき (**%Service_Console** または **%Service_Terminal** を使用) エンドユーザに秘密鍵を表示することを選択する場合、[**ユーザ編集**] ページの [**次回のログイン時にタイムベース・ワンタイム・パスワードの QR コードを表示する**] フィールドを選択することによってこの動作を有効にできます。これを行うと、ターミナル・セッションはエンドユーザの発行者、アカウント、および秘密鍵を表示します。Web アプリケーションは、エンドユーザの発行者、アカウント、および秘密鍵を QR コードと共に表示します。ここで、エンドユーザはコードをスキャンするか、情報を手入力できます。

  重要　　このオプションはお勧めしません。詳細は、以下の注意事項を参照してください。

注意　　以下は、2 要素の TOTP 認証を使用するときの重要なセキュリティ上の問題です。

- 安全でない環境では、秘密鍵や QR コードを転送しないでください。安全なネットワークでも帯域外転送をお勧めします (秘密鍵は InterSystems IRIS または InterSystems IRIS アプリケーションにログインする手段をエンドユーザに提供します。ユーザまたはエンドユーザが秘密鍵の安全性を確保できない場合、攻撃者がアクセスできるようになる可能性があり、秘密鍵がセキュリティの役に立たなくなります)。
- 組織に対して 2 要素の TOTP 認証を構成するとき、各エンドユーザに秘密鍵を直接渡すか、電話で伝えるか、または管理者が立ち会っている状況でエンドユーザに QR コードをスキャンさせることを強くお勧めします。こうすることによって、秘密鍵を取得する個人を認証する機会がもたらされます。

  ネットワークを介して秘密鍵を渡すと、漏えいの可能性が高くなります。これには、Web アプリケーション、コンソール、またはターミナルに最初にログインするときにエンドユーザに秘密鍵を表示することが含まれます。また、Web アプリケーションに最初にログインするときにエンドユーザに QR コードを表示することも含まれます。

図 B-2: TOTP 発行者、アカウント、鍵、および QR コード

注釈　　2 要素の TOTP 認証を使用していて、QR コードを生成する場合、InterSystems IRIS サーバで Java 1.7 以降を実行している必要があります。Java なしでも InterSystems IRIS で 2 要素の TOTP 認証を使用できますが、認証デバイスまたはアプリケーションでエンドユーザが発行者、アカウント、および鍵の値を手入力する必要があります。

サーバの 2 要素認証の構成

InterSystems IRIS サーバの 2 要素認証を構成する手順は以下のとおりです。

1. インスタンス全体に対する 2 要素認証の有効化および構成を行います。インスタンスを構成して SMS テキスト認証、TOTP 認証、またはその両方を使用できます。
2. SMS テキスト認証の場合、必要に応じて携帯電話サービス・プロバイダの構成を行います。これには以下が含まれます。
   - ある携帯電話サービス・プロバイダが必要であるのに、既定のプロバイダのリストに含まれていない場合、このサービス・プロバイダを追加する。
   - 既存のプロバイダに対し、必要に応じて構成情報を変更する (既定または追加)。

インスタンスに対する 2 要素認証設定の有効化および構成

InterSystems IRIS インスタンス (サーバ) 向けに 2 要素認証を設定すると、以下の 1 つまたは両方を有効にできます。

- **[2 要素のタイムベース・ワンタイム・パスワード認証]** (TOTP 認証)
- **[2 要素の SMS テキスト認証]**

いずれかの形式の 2 要素認証を有効にするための手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、**[認証/Web セッション・オプション]** ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[認証/Web セッション・オプション]**) に移動します。
2. 2 要素の TOTP 認証を有効にするには、**[認証/Web セッション・オプション]** ページで、**[2 要素のタイムベース・ワンタイム・パスワード認証を許可]** チェック・ボックスにチェックを付けます。**[2 要素のタイムベース・ワンタイム・パスワードの発行者]** フィールドが表示されます。ここにこの InterSystems IRIS インスタンスを識別する文字列を入力します。
3. 2 要素の SMS テキスト認証を有効にするには、**[認証/Web セッション・オプション]** ページで、**[2 要素の SMS テキスト認証を許可]** チェック・ボックスにチェックを付けます。以下の各フィールドが表示されます。
   - **[2要素タイムアウト (秒)]** – 1 回限りのセキュリティ・トークンを入力する際のタイムアウト秒数 (オプション)。
   - **[SMTPサーバのDNS名]** – SMS テキスト・メッセージの送信にこの InterSystems IRIS インスタンスが使用している SMTP (Simple Mail Transfer Protocol) サーバの DNS (Domain Name Service) 名 (**smtp.example.com** など) (必須)。
   - **[From (アドレス)]** – メッセージの “From” フィールドに表示されるアドレス (必須)。
   - **[SMTPユーザ名]** – (SMTP サーバが要求する場合) SMTP 認証のユーザ名 (オプション)。
   - **[SMTPパスワード]** および **[SMTPパスワード (確認)]** – SMTP 認証に対して入力および確認されるパスワード (オプション)。
4. **[保存]** をクリックします。
5. インスタンスが SMS テキスト認証をサポートしている場合、必要に応じて携帯電話サービス・プロバイダを構成します。これらの手順について、以下のセクションで説明します。

インスタンス自体に対してこのプロセスを完了した後、インスタンスのサービス、Web アプリケーション、クライアント・サーバ・アプリケーションなど、その他の構成を実行する必要があります。また、インスタンスのユーザを構成する必要があります。″2 要素認証の設定の概要″ にこれに関する一般的な指示が提示されています。

携帯電話サービス・プロバイダの構成

携帯電話サービス・プロバイダの構成に関連する項目は以下のとおりです。

- 携帯電話サービス・プロバイダの作成または編集
- 携帯電話サービス・プロバイダの削除
- 事前定義された携帯電話サービス・プロバイダ

携帯電話サービス・プロバイダの作成または編集

携帯電話サービス・プロバイダを作成または編集する手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、[**携帯電話サービスプロバイダ**] ページ ([**システム管理**] > [**セキュリティ**] > [**携帯電話**]) に移動します。
   - プロバイダを新規作成するには、[**新規プロバイダ**] をクリックします。
   - 既存のプロバイダを編集するには、プロバイダのテーブルにあるプロバイダの行で [**編集**] をクリックします。

   選択した携帯電話サービス・プロバイダの [**電話プロバイダ編集**] ページが表示されます。
2. [**電話プロバイダ編集**] ページで、以下のフィールドそれぞれの値を入力または変更します。
   - [**サービスプロバイダ**] – 携帯電話サービス・プロバイダの名前 (通常は会社名)。
   - [**SMSゲートウェイ**] – SMS (short message service) メッセージを送信するために携帯電話サービス・プロバイダが使用するサーバのアドレス。

携帯電話サービス・プロバイダの削除

携帯電話サービス・プロバイダを削除する手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、[**携帯電話サービスプロバイダ**] ページ ([**システム管理**] > [**セキュリティ**] > [**携帯電話**]) に移動します。
2. [**携帯電話サービスプロバイダ**] ページにあるプロバイダの行で [**削除**] をクリックします。
3. 削除の確認を求めるプロンプトが表示されたら、[OK] をクリックします。

事前定義された携帯電話サービス・プロバイダ

InterSystems IRIS には、事前定義された携帯電話サービス・プロバイダのリストが付属しており、それぞれのプロバイダには SMS (short message service) ゲートウェイが事前に設定されています。以下のとおりです。

- AT&T Wireless – **txt.att.net**
- Alltel – **message.alltel.com**
- Cellular One – **mobile.celloneusa.com**
- Nextel – **messaging.nextel.com**
- Sprint PCS – **messaging.sprintpcs.com**
- T-Mobile – **tmomail.net**
- Verizon – **vtext.com**

サービスに対する 2 要素認証の有効化または無効化

重要　**%Service_WebGateway** に対して、2 要素認証を一元的に有効または無効にすることができる場所はありません。"2 要素認証向けの Web アプリケーションの構成" の説明に従って、各アプリケーションに対して 2 要素認証を有効化または無効化します。

%Service_Bindings、**%Service_Console**、および **%Service _Terminal** に対して 2 要素認証を有効化または無効化する手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、[**サービス**] ページ ([**システム管理**]→[**セキュリティ**]→[**サービス**]) に移動します。

2. [**サービス**] ページで、いずれかの形式の 2 要素認証を有効にするサービスの名前をクリックします。選択したサービスの [**サービス編集**] ページが表示されます。
3. そのサービスの [**サービス編集**] ページで、[**2 要素の SMS**] チェック・ボックス、[**2 要素のタイムベース・ワンタイム・パスワード**] チェック・ボックス、またはその両方にチェックを付けるか、またはチェックを外します。これらの各チェック・ボックスは、そのインスタンスに対して 2 要素認証が有効になっている場合にのみ表示されます。
4. [**保存**] をクリックします。

2 要素認証向けの Web アプリケーションの構成

インスタンスに対して 2 要素認証を有効にしたら、これを使用するすべての Web アプリケーションでもこれを有効にする必要があります。アプリケーションでこれを有効にする手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、[**ウェブ・アプリケーション**] ページ ([**システム管理**]→[**セキュリティ**]→[**アプリケーション**]→[**ウェブ・アプリケーション**]) に移動します。
2. [**ウェブ・アプリケーション**] ページで、2 要素認証を有効にするアプリケーションの名前をクリックし、そのアプリケーションの [**編集**] ページを表示します。
3. [**編集**] ページの [**セキュリティ設定**] セクションで、[**2 要素の SMS**] チェック・ボックス、[**2 要素のタイムベース・ワンタイム・パスワード**] チェック・ボックス、またはその両方にチェックを付けるか、またはチェックを外します。これらの各チェック・ボックスは、そのインスタンスに対して 2 要素認証が有効になっている場合にのみ表示されます。

注釈　Web アプリケーションでは 2 要素認証と Web サービスの両方を同時にサポートすることはできません。

2 要素認証向けのエンドユーザの構成

2 要素認証向けに 1 回限りのセキュリティ・トークンを受信するようにエンドユーザを構成する手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、[**ユーザ**] ページ ([**システム管理**] > [**セキュリティ**] > [**ユーザ**]) に移動します。
2. 既存のユーザの場合は、編集するユーザの名前をクリックします。新規ユーザの場合は、[**新規ユーザ作成**] をクリックしてユーザの作成を始めます (新規ユーザの作成の詳細は、"新規ユーザの作成" を参照)。これらのアクションのどちらを実行しても、そのエンドユーザの [**編集**] ページが表示されます。
3. [**ユーザ編集**] ページで、[**SMS テキスト有効**] または [**タイムベース・ワンタイム・パスワード有効**] を適切に選択します。
4. [**SMS テキスト**] を選択した場合、以下のフィールドをすべて入力する必要があります。
   - [**携帯電話サービスプロバイダ**] – ユーザに携帯電話サービスを提供する会社。リストからプロバイダを選択します。プロバイダがリストに表示されていない場合は、[**新規プロバイダ**] をクリックして、InterSystems IRIS インスタンスに新規プロバイダを追加します([**新しいプロバイダーを作成**] をクリックすると、[**新しい携帯電話プロバイダーを作成**] ウィンドウが表示されます。このウィンドウには、[**サービスプロバイダ**] フィールドおよび [**SMSゲートウェイ**] フィールドがあります。これらの目的は、"携帯電話サービス・プロバイダの作成または編集" で説明した目的と同じです)。
   - [**携帯電話番号**] – ユーザの携帯電話番号。これは 2 つ目の要素で、1 回限りのセキュリティ・トークンを含むテキスト・メッセージをユーザが受信する電話番号です。
5. [**タイムベース・ワンタイム・パスワード有効**] を選択した場合、ページに以下のフィールドおよび情報が表示されます。
   - [**次回のログイン時にタイムベース・ワンタイム・パスワードの QR コードを表示する**] – ユーザが次回ログインしたときに QR コードを表示するかどうか。選択されている場合、InterSystems IRIS は、次回のログイン時に QR コードを表示し、これをスキャンして認証デバイスまたはアプリケーションに取り込むようユーザに求めます。さらに、トークンを表示して提供し、認証プロセスの完了を促します。既定では、このオプションは選択されていません。このオプションは使用しないことをお勧めします。

- **[新しいタイムベース・ワンタイム・パスワードの鍵を生成する]** – エンドユーザの新しい共有秘密と新しい QR コードの両方を作成して表示します。

  重要　ユーザに新しいタイムベース・ワンタイム・パスワードの鍵を生成すると、ユーザの認証アプリケーションの現在の鍵は機能しなくなります。ログインする前に、ユーザは、QR コードをスキャンするか、または手入力して新しい鍵を認証アプリケーションに入力する必要があります(これは既存のセッションには影響しません)。

- **[発行者]** – インスタンスの 2 要素の TOTP 認証を構成するときに設定した InterSystems IRIS インスタンスの識別子。
- **[アカウント]** – InterSystems IRIS アカウントの識別子。これはアカウントのユーザ名です。
- **[Base32 のタイムベース・ワンタイム・パスワード (OTP) の鍵]** – エンドユーザが認証デバイスまたはアプリケーションに入力する秘密鍵。
- **[QR コード]** – 発行者、アカウント、および秘密鍵の値が格納されているスキャン可能なコード。

6. **[保存]** をクリックして、このユーザのこれらの値を保存します。

サービスが 2 要素認証を使用していて、エンドユーザが 2 要素認証を有効にしている場合、認証には以下が必要です。

- SMS テキスト認証では、テキスト・メッセージを受信できる携帯電話。
- TOTP 認証では、検証コードを生成できるアプリケーションまたは認証デバイス。

これらがないと、エンドユーザは認証できません。

- SMS テキスト認証では、エンドユーザは携帯電話を所持していて、その電話でテキスト・メッセージを受信できる必要があります。これは、1 回限りのセキュリティ・トークンを含むテキスト・メッセージを SMS テキストとしてユーザが受信する電話番号です。
- TOTP 認証では、ユーザは、QR コードをスキャンできるか、または TOTP (検証コードとして作用) の生成に必要な秘密鍵などの情報を受け入れることができる、認証デバイスまたはアプリケーションを所持している必要があります。

2 要素認証向けのバインディング・クライアントの構成

クライアント・サーバ接続は **%Service_Bindings** を使用します。この接続で 2 要素認証を使用するために必要なコードは、プログラミング言語によって異なります(コンソール、ターミナル、および Web アプリケーションではクライアント側の構成は必要ありません)。サポートされている言語には以下のものがあります。

- Java および JDBC
- .NET
- ODBC

クライアント側のコードで実行される処理は以下の 3 つです。

1. InterSystems IRIS サーバへの接続を確立した後、2 要素認証がサーバで有効になっているかどうかを確認します。通常、この確認にはクライアントの接続オブジェクトのメソッドが使用されます。
2. ユーザから 1 回限りのセキュリティ・トークンを取得します。この処理では一般的に、InterSystems IRIS と特別な関係のないユーザ・インタフェース・コードが使用されます。
3. 1 回限りのセキュリティ・トークンを InterSystems IRIS サーバに提供します。この処理でも一般的に、接続オブジェクトのメソッドが使用されます。

注釈　ユーザが **%Service_Bindings** を使用してログインすると、InterSystems IRIS はスキャンする QR コードを表示しません。ユーザは、認証デバイスまたはアプリケーションを事前に設定しておく必要があります。

重要　**%Service_Bindings** を使用して InterSystems IRIS サーバに接続するスタジオは、2 要素認証をサポートしていません。

Java および JDBC

Java では、2 要素認証のサポートは、以下のように **IRISConnection** クラスのメソッドを 2 つ使用します。

. `public boolean isTwoFactorEnabled() throws Exception`

  このメソッドは、サーバで 2 要素認証が有効になっているかどうか確認します。このメソッドはブーリアン値を返します。`true` は、2 要素認証が有効であることを意味します。

. `public void sendTwoFactorToken(String token) throws Exception`

  このメソッドは、1 回限りのセキュリティ・トークンをサーバに提供します。このメソッドは 1 つの引数 token をとります。これは、ユーザが受信した 1 回限りのセキュリティ・トークンです。

以下の例では、conn という接続のインスタンスを使用します。

1. この例では、このインスタンスのメソッドを使用して、2 要素認証が有効になっているかどうか確認します。
2. この例では、サーバにトークンを提供しようとし、これに失敗するとエラー処理が行われます。

重要　2 要素認証がサーバで有効になっていて、クライアント・コードが 2 要素認証呼び出しを実装していない場合、サーバはクライアントとの接続を切断します。

```
// Given a connection called "conn"
if (conn.isTwoFactorEnabled()) {
  // Prompt the user for the two-factor authentication token.
  // Store the token in the "token" variable.
  try {
    conn.sendTwoFactorToken(token);
  }
  catch (Exception ex) {
    // Process the error from a invalid authentication token here.
  }
}
```

.NET

.NET の場合、InterSystems IRIS は、Managed Provider および ADO.NET との 2 要素認証を使用した接続をサポートします。2 要素認証のサポートは、以下のように **tcp_conn** クラスのメソッドを 2 つ使用します。

- `bool IRISConnection.isTwoFactorEnabledOpen()`

  このメソッドは、InterSystems IRIS サーバへの接続を開き、2 要素認証がこの接続で有効になっているかどうかを確認します。このメソッドはブーリアン値を返します。`true` は、2 要素認証が有効であることを意味します。

- `void IRISConnection.sendTwoFactorToken(token)`

  このメソッドは、1 回限りのセキュリティ・トークンをサーバに提供します。返り値はありません。このメソッドは 1 つの引数 token をとります。これは、ユーザが受信した 1 回限りのセキュリティ・トークンです。トークンの問題 (有効でないなど) と接続の問題のいずれかがある場合、このメソッドは例外をスローします。

重要　クライアント・アプリケーションは、IRISConnection.Open を呼び出す代わりに isTwoFactorEnabledOpen を呼び出します。isTwoFactorEnabledOpen メソッドは、続けて sendTwoFactorToken を呼び出す必要があります。

また、2 要素認証がサーバで有効になっていて、クライアント・コードが 2 要素認証呼び出しを実装していない場合、サーバはクライアントとの接続を切断します。

以下の例では、conn という接続のインスタンスを使用します。

1. この例では、このインスタンスのメソッドを使用して、2 要素認証が有効になっているかどうか確認します。
2. この例では、サーバにトークンを提供しようとし、これに失敗するとエラー処理が行われます。

```
// Given a connection called "conn"
try {
  if (conn.isTwoFactorEnabledOpen()) {
    // Prompt the user for the two-factor authentication token.
    // Store the token in the "token" variable.
    conn.sendTwoFactorToken(token);
  }
}
catch (Exception ex) {
  // Process exception
}
```

ODBC

ODBC では、2 要素認証のサポートは、ODBC の標準的な関数呼び出しを 2 つ使用します (″Microsoft ODBC API リファレンス″ を参照)。

. `SQLRETURN rc = SQLGetConnectAttr(conn, 1002, &attr, sizeof(attr), &stringLengthPtr);`

  Microsoft ODBC API の一部である SQLGetConnectAttr 関数は、指定された接続属性の現在値を返します。InterSystems ODBC クライアントは、この関数を使用して、サーバが 2 要素認証をサポートしているかどうかを判定します。最初の引数の値は、クライアントからサーバへの接続のハンドルです。2 つ目の引数の値は、`1002` で、これは 2 要素認証がサポートされているかどうかを指定する ODBC 属性です。これ以降の引数の値は、属性 1002 の値を含む文字列および関係のある変数のサイズ用です。

. `SQLRETURN rc = SQLSetConnectAttr(conn, 1002, securityToken, SQL_NTS);`

  同じく Microsoft ODBC API の一部である SQLSetConnectAttr 関数は、指定された接続属性の値を設定します。InterSystems ODBC クライアントは、この関数を使用して、2 要素認証トークンの値をサーバに送信します。4 つの引数の値は、それぞれ以下のとおりです。

  - クライアントからサーバへの接続。
  - `1002`。2 要素認証がサポートされているかどうかを指定する ODBC 属性です。
  - 1 回限りのセキュリティ・トークンの値。
  - `SQLNTS`。1 回限りのセキュリティ・トークンが文字列に格納されていることを示します。

重要　2 要素認証がサーバで有効になっていて、クライアント・コードが 2 要素認証呼び出しを実装していない場合、サーバはクライアントとの接続を切断します。

以下の例では、conn という接続のインスタンスを使用します。

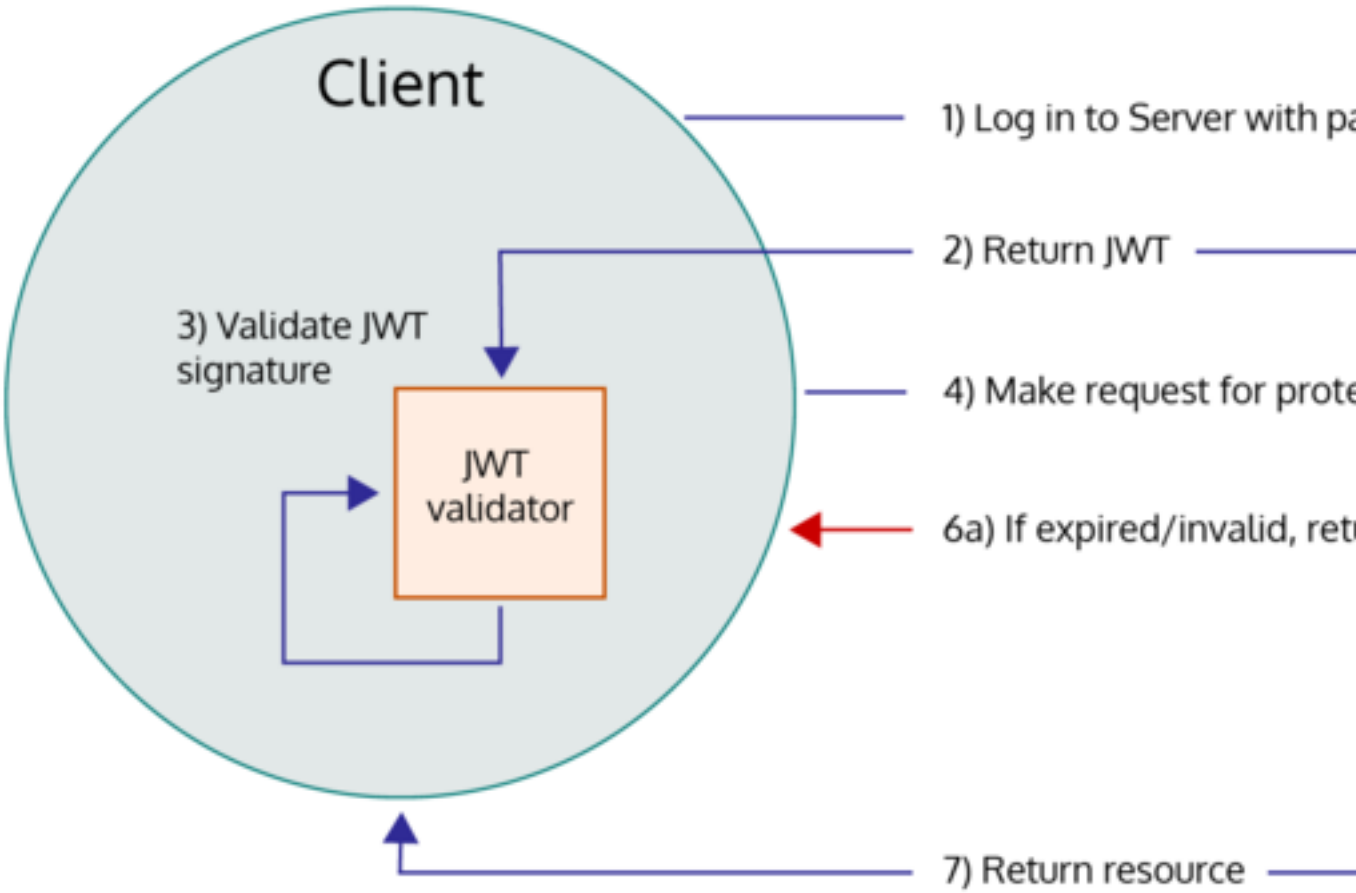
1. この例では、SQLGetConnectAttr を使用して、2 要素認証が有効になっているかどうか確認します。
2. この例では、SQLSetConnectAttr 呼び出しによってサーバにトークンを提供しようとし、これに失敗するとエラー処理が行われます。SQLSetConnectAttr が失敗すると、サーバは接続を切断します。そのため、再び認証を行うには、その前に接続を再確立する必要があります。

```
// Given a connection called "conn"
SQLINTEGER stringLengthPtr;
SQLINTEGER attr;
SQLRETURN rc = SQLGetConnectAttr(conn, 1002, &attr, sizeof(attr), &stringLengthPtr);
if attr {
  // Prompt the user for the two-factor authentication token.
  wstring token;
  SQLRETURN rc = SQLSetConnectAttr(conn, 1002, token, SQL_NTS);
    if !rc {
      // Process the error from a invalid authentication token.
    }
}
```

JSON Web トークン (JWT) 認証

JWT 認証の概要

JWT は、コンパクトで URL セーフな、認証、承認、または情報交換の手段です。認証の場合、サーバは既に認証されているクライアントに JWT を提供し、この JWT の期限が切れるまでは、これらのクライアントがサーバ上の保護されたリソースにアクセスするのに再度パスワードを提供しなくてもよいようにします。認証フローは以下の図のようになります。



JWT 認証の構成

JWT 認証を使用するには、クライアントがその整合性を検証できるよう、InterSystems IRIS に JWT を発行するための構成が必要です。

- **[システム管理]→[セキュリティ]→[システム・セキュリティ]→[認証/ウェブセッションオプション]** ページには、以下の設定が含まれています。
  - **[JWT 発行者フィールド]** ー クライアントに対する JWT の発行者を特定する JWT ペイロード内の `iss` クレームを定義します。
  - **[JWT 署名アルゴリズム]** ー InterSystems IRIS が JWT の署名および検証に使用する 暗号化アルゴリズムを決定します。
  - **[キーストアをリセット]** ー 暗号化キーをローテーションさせます。これにより、以前発行した期限の切れていないすべての JWT が無効になります。

必要な構成は、JWT の発行だけではありません。JWT を受け入れ、使用するには、これを受け取るクライアントを構成する必要があります。REST Web アプリケーションは、認証に JWT を使用できるクライアントの 1 つです。これらの設定については、"アプリケーションの作成および編集" ページを参照してください。

JWT 使用のサポート

InterSystems IRIS は、REST API および OAuth 2.0 フレームワークでの認証のための JWT の使用をサポートしています。

REST API の使用

JWT 認証用に構成を行うと、REST API はディスパッチ・クラスの **UrlMap** に含めるべきではない 4 つのエンドポイントを取得します。

- `/login` ー HTTP 基本認証または要求本文内の有効な認証情報を使用してこのエンドポイントを呼び出すと、後続の要求で使用できるアクセス・トークンとリフレッシュ・トークンが返されます。
- `/logout` ー Group-By-ID を使用していない場合にこのエンドポイントを呼び出すと、提供されたアクセス・トークンと関連付けられたリフレッシュ・トークンが無効になります。Group-By-ID を使用している場合は、現在の By-ID グループを持つすべてのセッションが無効になります。
- `/refresh` ー このエンドポイントの呼び出しは、有効なリフレッシュ・トークンを使用して呼び出すと、新しいアクセス・トークンとリフレッシュ・トークンのペアが発行されます。これにより、以前のアクセス・トークンとリフレッシュ・トークンのペアは無効になります。
- `/revoke` ー Group-By-ID を使用していない場合、これは機能的に `/logout` と同じです。Group-By-ID を使用している場合は、現在のアクセス・トークンとリフレッシュ・トークンのペアのみが取り消されます。

ディスパッチ・クラスのエンドポイント名は、以下を使用してカスタマイズできます。

```
Parameter TokenLoginEndpoint = "mylogin";
Parameter TokenLogoutEndpoint = "mylogout";
Parameter TokenRevokeEndpoint = "myrevoke";
Parameter TokenRefreshEndpoint = "myrefresh";
```

JWT を使用した REST エンドポイントへのアクセス

HTTP 要求の REST API へのアクセス・トークンは、ヘッダ内に `Authorization: Bearer ACCESS_TOKEN_HERE` の形式を使用して指定します。要求で認証情報の代わりにこのアクセス・トークンを指定すること以外、`/login` エンドポイントと `/refresh` エンドポイントを除き、通常どおり Web アプリケーション・エンドポイントにアクセスします。アクセス・トークンを取得するには、まず、`/login` エンドポイントにアクセスします。

`/login` エンドポイント

`/login` エンドポイントにアクセスし、アクセス・トークンとリフレッシュ・トークンを取得するには、認証ヘッダは含めず、本文に JSON 形式で以下のように認証情報を含めて、HTTP POST 要求を作成します。

```
{"user": "YOUR USER",
"password": "YOUR PASSWORD"}
```

認証情報が有効な場合、以下のような応答を受け取ります。

```
{
"access_token":
"[illegible]",
"refresh_token":
"[illegible]",
"sub":
"YOUR USER",
"iat":
1682707417.749942,
"exp":
1682707477
}
```

/login アクセス・トークンを例として使用すると、別の REST API エンドポイントへの要求の Authorization ヘッダには、以下のような値が含まれます。

```
Bearer
eyJ[illegible]
```

/refresh エンドポイント

アクセス・トークンなしで、HTTP POST 要求を使用して、/refresh エンドポイントにアクセスします。代わりに、要求の本文で以下の JSON 形式のデータを送信します。

```
{
"refresh_token": "YOUR REFRESH TOKEN",
"grant_type":
"refresh_token"
}
```

これにより、/login エンドポイントへのアクセスと同様に、新しいアクセス・トークンとリフレッシュ・トークンのペアが返されますが、ログアウトによってセッションが失われることはありません。

OAuth 2.0 および OpenID Connect の使用

InterSystems IRIS での OAuth 2.0 フレームワークを使用した JWT のサポートの詳細は、"OAuth 2.0 および OpenID Connect の使用法" を参照してください。

LDAP

LDAP と InterSystems IRIS®

InterSystems IRIS® は、LDAP (Lightweight Directory Access Protocol) を使用した認証および承認をサポートします。LDAP システムには、InterSystems IRIS が情報を取得するユーザ情報の一元管理リポジトリがあります。例えば Windows の場合、Active Directory を使用するドメイン・コントローラが LDAP サーバになります。

以下がサポートされます。

- LDAP 認証 – InterSystems IRIS により、ユーザにユーザ名とパスワードの入力を求めるプロンプトが表示されます。インスタンスは LDAP サーバに関連付けられ、LDAP サーバは認証を実行し、ユーザのロールおよびその他の承認情報を取得します。インスタンスが LDAP サーバに接続できない場合に、キャッシュ認証情報を使用してユーザを認証するようインスタンスを構成することもできます。
- LDAP 承認 – インターシステムズは LDAP グループをサポートしており、承認の一部としてロールを指定できます。ローカルの InterSystems IRIS ターミナルでは、LDAP 承認と OS ベースの認証が併用されます(ターミナルへのアクセスは、Windows では **%Service_Console** によって、他のすべてのオペレーティング・システムでは **%Service_Terminal** によって管理されます)。

InterSystems IRIS では、同時に複数の LDAP ドメインに認証と承認を提供することもできます。

LDAP を InterSystems IRIS の代行認証機能と併用することもできます。これにより、カスタム・メカニズムを実装して、インターシステムズのセキュリティに含まれる認証とロール管理アクティビティを置き換えることができます。

InterSystems IRIS では以下に対する LDAP サポートが提供されます。

- Active Directory
- OpenLDAP
- LDAP バージョン 3 プロトコル (これより前の LDAP プロトコルはサポートされていません)

LDAP 認証

LDAP 認証の設定の概要

認証に LDAP サーバを使用するように InterSystems IRIS のサービスまたはアプリケーションを構成する手順は以下のとおりです。

1. LDAP サーバを使用するように InterSystems IRIS を構成します。
   a. インスタンスで LDAP および関連機能を有効化します。
   b. InterSystems IRIS のインスタンスで使用する LDAP 構成を作成します。ここで、InterSystems IRIS ユーザのプロパティの値を設定するために使用する LDAP ユーザ・プロパティの名前を指定します。
   c. オプションで、LDAP 構成をテストします。
   d. オプションで、複数の LDAP ドメインをサポートするようにインスタンスを構成します。
   e. インスタンスへのログインに必要なロールを設定します。
   f. インスタンスの関連サービスおよびアプリケーションで LDAP を有効化します。この設定では、InterSystems IRIS のインスタンス全体で LDAP を有効にして、関連するサービスやアプリケーションに対して LDAP を有効にします。

注釈　プログラムで LDAP 認証を実行するには、InterSystems IRIS の代行認証を使用します。

インスタンスでの LDAP の有効化

LDAP を使用するように InterSystems IRIS のインスタンスを構成するには、まず、使用する機能を有効にします。

1. 管理ポータルのホーム・ページで、[**認証/Web セッション・オプション**] ページ ([**システム管理**] > [**セキュリティ**] > [**システム・セキュリティ**] > [**認証/Web セッション・オプション**]) に移動します。
2. [**認証/Web セッション・オプション**] ページで、以下の手順を実行します。
   - LDAP 認証を有効にするには、[**LDAP認証を許可**] を選択します。
   - LDAP キャッシュ資格情報を使用した認証を有効にするには、[**LDAPキャッシュ credentials 認証を許可**] を選択します。このトピックの詳細は、"LDAP キャッシュ認証情報" を参照してください。
3. [**保存**] をクリックすると、変更内容が適用されます。

LDAP キャッシュ認証情報

LDAPキャッシュ認証情報を使用するようにインスタンスを構成すると、そのインスタンスでは、各ユーザの認証に前回使用した認証情報のコピーが保存(キャッシュ)されます。インスタンスでキャッシュ認証情報がサポートされている場合に、そのインスタンスが LDAP サーバに接続できないときは、キャッシュされている LDAP 認証情報を使用してユーザを認証します。この状況は、LDAP サーバ自体または LDAP サーバへの接続に問題がある場合に生じる可能性があります。

キャッシュ認証情報を保護するために、InterSystems IRIS では、すべての LDAP パスワードを単方向ハッシュとしてセキュリティ・データベースに格納します。インスタンスが LDAP サーバを使用してユーザを検証できない場合、以下のことを確認しようとします。

- 入力されたパスワードのハッシュが、格納されているパスワードのハッシュと一致している
- 前回の LDAP ログインからのキャッシュされた有効期限に達していない

両方の条件を満たす場合、インスタンスはユーザを認証して、ログインが続行します。それ以外の場合、ログインは失敗します。

LDAP 構成の作成または変更

LDAP 認証を実行するために、InterSystems IRIS は LDAP 構成を使用します。LDAP 構成では、特定のセキュリティ・ドメインに関する LDAP サーバへの接続を指定し、以下の処理を行うために必要な情報を定義します。

- LDAP サーバに接続し、クエリを実行します。
- 認証するユーザについて必要な情報を取得します。

注釈　インスタンスに対して Kerberos が有効になっている場合、LDAP 構成のすべてのメニュー項目と他のラベルは LDAP/Kerberos 構成を指します。以下の手順では、この点について状況ごとに個別には注記していません。

LDAP 構成を作成または変更する手順は以下のとおりです。

1. 管理ポータルの [**セキュリティ LDAP 構成**] ページ ([**システム管理**] > [**セキュリティ**] > [**システム・セキュリティ**] > [**LDAP 構成**]) に移動します。

   インストール時に、現在 LDAP サーバを使用しているマシンに InterSystems IRIS をインストールすると、LDAP サーバのドメインおよびその他の構成情報に基づいて LDAP 構成が作成されます。
2. 構成を作成または変更します。
   - 既存の構成を変更するには、その名前をクリックします。例えば、ローカル LDAP サーバに関連付けられた構成を使用する場合、その構成の属性をチェックし、必要に応じて変更します。
   - 構成を作成するには、[**新規 LDAP 構成の作成**] ボタンをクリックします。[**LDAP 構成の編集**] ページが表示されます。

注釈　構成の作成時には、[LDAP 構成の編集] ページの [LDAP 構成] チェック・ボックスにチェックを付けます (使用可能な場合)。LDAP 構成を定義するフィールドが表示されます。

3. このフィールドを変更するか、値を入力して、構成を定義します (以下に表示)。
4. 複数の構成を作成する場合は、[システムワイドセキュリティパラメータ] ページ ([セキュリティ管理] > [セキュリティ] > [システム・セキュリティ] > [システムワイドセキュリティパラメータ]) で [デフォルトセキュリティドメイン] ドロップダウンを使用して、既定のものを指定する必要があります。

LDAP 構成フィールド

LDAP 構成には以下のフィールドがあります。

- [ログインドメイン名] ― 必須。LDAP 構成の名前です。これは通常、example.com や example.org のような形式です。

  ピリオドを含まない値を入力した場合、.com が自動的に追加されるため、example は example.com になります。値を大文字で入力した場合は自動的に小文字になるため、EXAMPLE.COM は example.com になります。両方の変換がシステムによって適宜実行されます。

  [名前] フィールドの変換後の値を使用して、[検索に使用するLDAPベースDN] フィールドに値が入力されます。
- [説明] ― 構成を説明する任意のテキスト。
- [コピー元] ― 構成の作成時にのみ使用できます。この LDAP 構成の初期値を指定するために InterSystems IRIS が既存の LDAP 構成から属性をコピーするかどうか。
- [LDAP 有効] ― InterSystems IRIS が LDAP サーバに接続するためにこの構成を使用できるかどうか。
- [LDAP サーバは Windows Active Directory サーバ] ― Windows のみ。LDAP サーバが Windows Active Directory サーバであるかどうか。
- [LDAP ホスト名] ― 必須。LDAP サーバが動作しているホストの名前。ホスト名の複雑さは、未修飾のホスト名から、ポート番号を持つ完全修飾ホスト名まで多岐にわたります。ホスト名の必要な形式は、その構成により異なります。

  LDAP サーバが特定のポートを使用するように構成されている場合、ホスト名に “:portname” を追加してそれを指定できます。一般的には、ポートは指定せず、LDAP 機能で既定のポートを使用するようにします。ネットワークに複数の複製されたドメイン・サーバがある場合は、ホスト名としてドメインの `example.com` を以下のように指定できます。

  ```
  ldapserver.example.com
  ldapserver1.example.com
  ldapserver2.example.com
  ldapserver3.example.com
  ```

  LDAP は一致するすべての LDAP サーバのアドレスについて DNS クエリを実行し、接続する LDAP サーバを自動的に選択します。

重要　[LDAPホスト名] の値にポート番号を含めると、接続を確立する際の TLS の動作を制御できます。

- 指定した値に 636 以外のポート番号が含まれていて (`ldapserver.example.com:389` など)、[LDAPセッションに TLS/SSL 暗号化を使用する] チェック・ボックスにチェックが付いている場合、そのインスタンスは LDAP サーバとのプレーン・テキスト接続を確立し、StartTLS コマンドを発行して接続を暗号化しようとします。
- LDAP ホスト名に指定した値にポート番号 636 が含まれている場合 (`ldapserver.example.com:636` など)、そのインスタンスは、[LDAPセッションに TLS/SSL 暗号化を使用する] チェック・ボックスにチェックが付いているかどうかに関係なく、LDAP サーバとの TLS 接続を確立しようとします。ただし、UNIX® クライアント・インスタンスからポート 636 に直接接続することはサポートされていません。

背景情報については、%SYS.LDAP.Init() メソッドのクラス・リファレンスを参照してください。

- LDAP 検索情報 ― 状況により異なります。
  - [検索に使用するLDAPユーザ名] ― Windows Active Directory サーバの場合のみ。使用可能な場合は必須。初期接続を確立し、LDAP 検索の実行に使用するために LDAP サーバに提供されるユーザ名です。このユーザは検索ユーザとも呼ばれます。

    検索ユーザには、LDAP データベース全体の読み取り許可が必要です。検索ユーザの LDAP データベースへのアクセスが遮られないようにすることが重要です。例えば、ユーザの LDAP アカウントは、以下のように設定する必要があります。

    - ユーザがアカウントのパスワードを変更できない
    - パスワードの有効期限を無期限にする
    - アカウントの有効期限を無期限にする

    LDAP データベースでの検索の詳細は、"LDAP データベース内でターゲット・ユーザが検索される仕組み" を参照してください。
  - [LDAP 検索ユーザー DN] ― Windows 以外のすべてのプラットフォームおよび Windows Active Directory 以外のサーバの場合。使用可能な場合は必須。初期接続を確立し、LDAP 検索の実行に使用するために LDAP サーバに提供されるユーザの識別名 (DN) です。このユーザは検索ユーザとも呼ばれます。

    検索ユーザには、LDAP データベース全体の読み取り許可が必要です。検索ユーザの LDAP データベースへのアクセスは遮られないようにすることが重要です。例えば、ユーザの LDAP アカウントは、以下のように設定する必要があります。

    - ユーザがアカウントのパスワードを変更できない
    - パスワードの有効期限を無期限にする
    - アカウントの有効期限を無期限にする

    例えば、検索ユーザが “ldapsearchuser” である場合、LDAP DN (識別名) は以下のようになります。

    ```
    uid=ldapsearchuser,ou=People,dc=example,dc=com
    ```

    LDAP データベースでの検索の詳細は、"LDAP データベース内でターゲット・ユーザが検索される仕組み" を参照してください。
- [LDAPユーザ名パスワード] ― 構成の作成時または変更時にのみ使用可能。最初の接続に使用するアカウントに関連付けられたパスワード。
- [検索に使用するLDAPベースDN] ― 必須。検索を開始する起点となるディレクトリ・ツリー内のポイント。通常は、`DC=example,DC=com` のように、ドメイン・コンポーネントで構成されます。

- [LDAP Base DN to use for Nested Groups searches] — 必須。ネストしたグループの検索を開始する起点となるディレクトリ・ツリー内のポイント。一般的に組織単位とドメインの要素で構成され、`OU=IRIS,OU=Groups,DC=test,DC=com` のようになります。既定では、[検索に使用するLDAPベースDN] と同じ値に設定されています。
- [LDAPユニーク検索属性] — 必須。各レコードの一意の識別要素。これにより、適切な検索が行われます。LDAP データベースでの検索の詳細は、"LDAP データベース内でターゲット・ユーザが検索される仕組み" を参照してください。
- [LDAPセッションに TLS/SSL 暗号化を使用する] — InterSystems IRIS インスタンスと LDAP サーバが TLS を使用して通信を暗号化するかどうか (既定では無効)。

  重要　　LDAP に対して TLS 暗号化を有効にすることをお勧めします。

  Active Directory サーバへの接続の場合は、以下の点に注意してください。

  - この設定を Windows 上のインスタンスから Active Directory サーバへの LDAP 接続で有効にした場合、接続はポート 636 (TLS で暗号化されたポート) を使用します。
  - この設定を UNIX® 上のインスタンスから Active Directory サーバへの LDAP 接続で有効にした場合、InterSystems IRIS は最初にポート 389 (暗号化されていない LDAP ポート) で接続を確立し、その後、StartTLS 呼び出しによって暗号化が有効になります。

  Active Directory サーバで LDAP server signing requirements パラメータを `Require signature` に設定することもお勧めします。これにより、チャンネルが StartTLS で暗号化されている場合を除き、サーバのポート 389 で LDAP bind コマンドが実行されるのを防ぐことができます。詳細は、Microsoft Web サイトのドメイン コントローラーの LDAP サーバー署名要件の記事を参照してください。
- [File with Certificate Authority certificate(s) to authenticate the LDAP server] — UNIX® のみ。サーバの認証に使用される TLS 証明書 (PEM 形式) を含むファイルの場所。

  Windows で、サーバ証明書の認証に使用される TLS 証明書 (PEM 形式) を含むファイルの場所を指定して、安全な LDAP 接続を確立するには、Microsoft Certificate Services を使用します。証明書は、Certificates (Local Computer)¥Trusted Root Certification Authorities 証明書ストアにインストールする必要があります。
- [ISC_LDAP_CONFIGURATION 環境変数を許可] — OS ベースの LDAP および複数のドメインを使用する場合、ISC_LDAP_CONFIGURATION 環境変数を使用するかどうかを指定します。この環境変数が定義されている場合、OS ベースの LDAP は、これを使用して、認証に使用する LDAP 構成を特定します。
- [ロール、ルーチン、ネームスペースで LDAP グループを使用する] — ユーザのロール、ルーチン、およびネームスペースが、ユーザのグループ・メンバシップから取得されるかどうか (既定では True です)。グループ・メンバシップから取得されない場合、ユーザの LDAP レコードの属性フィールドから取得されます。このフィールドを選択すると、他のフィールドが有効または無効になります (詳細は、後続の各フィールドを参照してください)。

  注釈　　承認には LDAP 属性 (登録されている LDAP プロパティを含む) ではなく、LDAP グループの使用をお勧めします。既存のコードがある、または登録されているプロパティを使用する必要がある場合、詳細は "LDAP 属性を使用した承認の構成" を参照してください。

- [ネストしたグループのロール/ルーチン/ネームスペースを検索] — [LDAP サーバが Windows アクティブ・ディレクトリ・サーバ] および [ロール/ルーチン/ネームスペースに LDAP グループを使用] が選択されている場合にのみ有効です。ユーザの入れ子になったグループすべてを検索で返すかどうか。入れ子になったグループの詳細は、"入れ子になったグループ" を参照してください。
- [グループ名の組織IDプレフィックス] — [ロール/ルーチン/ネームスペースに LDAP グループを使用] が選択されている場合にのみ有効です。 詳細は、"LDAP グループ名の構成" を参照してください。

- **[ユニバーサルグループ承認を許可する]** – **[ロール/ルーチン/ネームスペースに LDAP グループを使用]** が選択されている場合にのみ有効です。すべての InterSystems IRIS インスタンスに関連する、LDAP サーバ上の属性を検索で使用するかどうか。詳細は、"ユニバーサル LDAP 承認グループの作成" を参照してください。
- **[承認グループID]** – **[ロール/ルーチン/ネームスペースに LDAP グループを使用]** が選択されている場合にのみ有効です。このインスタンスが属する複数インスタンス・グループ。詳細は、"複数のインスタンスを対象とする LDAP 承認グループ (複数インスタンス・グループ) の作成" を参照してください。
- **[承認インスタンスID]** – **[ロール/ルーチン/ネームスペースに LDAP グループを使用]** が選択されている場合にのみ有効です。このインスタンスが属する単一インスタンス・グループ。詳細は、"1 つのインスタンスを対象とする LDAP 承認グループ (単一インスタンス・グループ) の作成" を参照してください。
- **[デフォルトネームスペースを取得するためのユーザ属性]** (LDAP グループが選択されている場合は無効) – 値がユーザの Startup namespace プロパティのソースである属性。InterSystems IRIS ユーザのこのプロパティについては、"ユーザ・アカウントのプロパティ" で説明されています。この LDAP プロパティの詳細は、"LDAP 属性を使用した承認の構成" を参照してください。
- **[デフォルトルーチンを取得するためのユーザ属性]** (LDAP グループが選択されている場合は無効) – 値がユーザの Tag^Routine プロパティのソースである属性。InterSystems IRIS ユーザのこのプロパティについては、"ユーザ・アカウントのプロパティ" で説明されています。この LDAP プロパティの詳細は、"LDAP 属性を使用した承認の構成" を参照してください。
- **[ロール取得のためのユーザ属性]** (LDAP グループが選択されている場合は無効) – その値によってユーザの割り当て先のロールを決定する属性。この属性は LDAP 複数値属性として指定し、作成する必要があります。InterSystems IRIS ユーザのロールの詳細は、ユーザの **[ユーザ編集]** ページの **[ロール]** タブを参照してください。この LDAP プロパティについては、"LDAP 属性を使用した承認の構成" に説明があります。
- **[コメント属性を取得するためのユーザ属性]** – 値がユーザの Comment プロパティのソースである属性。このプロパティの詳細は、"ユーザ・アカウントのプロパティ" を参照してください。ユーザがログインすると、Security.Users.Get() メソッドを使用して、このプロパティの値を取得できます。
- **[フルネームを取得するためのユーザ属性]** – 値がユーザの Full name プロパティのソースである属性。このプロパティの詳細は、"ユーザ・アカウントのプロパティ" を参照してください。ユーザがログインすると、Security.Users.Get() メソッドを使用して、このプロパティの値を取得できます。
- **[メールアドレスを取得するユーザー属性]** – 値がユーザの Email address プロパティのソースである属性。このプロパティの詳細は、"ユーザ・アカウントのプロパティ" を参照してください。ユーザがログインすると、Security.Users.Get() メソッドを使用して、このプロパティの値を取得できます。
- **[携帯電話番号を取得するユーザー属性]** – 値がユーザの Mobile Phone Number プロパティのソースである属性。このプロパティの詳細は、"ユーザ・アカウントのプロパティ" を参照してください。ユーザがログインすると、Security.Users.Get() メソッドを使用して、このプロパティの値を取得できます。
- **[携帯電話プロバイダを取得するためのユーザ属性]** – 値がユーザの Mobile Phone Service Provider プロパティのソースである属性。このプロパティの詳細は、"ユーザ・アカウントのプロパティ" を参照してください。ユーザがログインすると、Security.Users.Get() メソッドを使用して、このプロパティの値を取得できます。
- **[各ユーザに取得するLDAP属性]** – 値がアプリケーション固有の変数のソースである属性。これにより、アプリケーション・コードで Security.Users クラスの Get メソッドを使用してこの情報が返されます。

LDAP 構成の各フィールドの値は、Security.LDAPConfigs クラスのインスタンスに保存されます。

LDAP/Kerberos 構成フィールドに関する注意事項

インスタンスに対して Kerberos 認証が有効になっている場合、LDAP 構成を作成するページは、**[LDAP/Kerberos 構成を編集]** ページです。このページには、**[LDAP 構成を編集]** ページと同じフィールドがあります。"LDAP 構成フィールド" を参照してください。

LDAP 構成のテスト

LDAP 構成を作成したら、それをテストできます。これにより、LDAP サーバに適切に接続できることを確認することや、問題が発生する場合はトラブルシューティングすることができます。構成をテストする手順は以下のとおりです。

1. 管理ポータルで、[セキュリティ LDAP 構成] ページ ([システム管理] > [セキュリティ] > [システム・セキュリティ] > [LDAP 構成]) に移動します。
2. [LDAP 認証のテスト] をクリックします。
3. [ユーザ名] フィールドと [パスワード] フィールドに、LDAP サーバで定義されている有効なユーザ名とパスワードを入力します。複数のドメインを使用するようにインスタンスが構成されている場合は、`EndUser@example.com` のように、完全修飾ユーザ名を入力する必要があります。インスタンスが 1 つのドメインのみを使用している場合は、`EndUser` のように、単に未修飾のユーザ名 (@ 記号やドメイン名なし) を入力します。
4. [テスト] をクリックします。

[テスト結果] フィールドに、LDAP サーバからの出力が表示されます。

注釈　この機能は、インスタンスが LDAP サーバに接続して、入力されたユーザの認証チェックを実行できるかどうかのみをテストします。承認チェックまたは許可チェックを実行して、ユーザがシステムに正常にログインできるかどうかは確認しません。

入力されたユーザのテストは成功したのにユーザがログインできない場合は、ログイン・エラーがないかどうか監査レコードを確認してください。正常にログインするために、ユーザに追加の許可を付与しなければならないことがあります。

複数の LDAP ドメインの使用

InterSystems IRIS では、複数のドメインでの LDAP 認証がサポートされています。これにより、EndUser@example.com や EndUser@otherexample.com など、複数のドメインに属する同じユーザ名を含むユーザ・アカウントをインスタンスで保持できます。この機能は、さまざまなシナリオで役立ちます。以下に例を示します。

- 各ユーザの一意の識別子を保持しながら、複数のドメインの個々のユーザ・セットを 1 つの大きなグループにマージできます。
- それぞれ特権が異なるアカウントを同じ個人が複数のドメインに持つことができます。

複数のドメインを使用する手順は以下のとおりです。

1. "LDAP 構成の作成または変更" で説明した手順に従って、追加の LDAP 構成を作成します。
2. 複数のドメインを使用するようにインスタンスを構成し、既定のドメインを指定します。
   a. インスタンスで複数のドメインの使用を有効にします。管理ポータルの [システムワイドセキュリティパラメータ] ページ ([システム管理] > [セキュリティ] > [システム・セキュリティ] > [システムワイドセキュリティパラメータ]) で、[複数セキュリティドメインを許可する] チェック・ボックスにチェックを付けます。
   b. 既定のドメインを指定します。管理ポータルの [システムワイドセキュリティパラメータ] ページ ([システム管理] > [セキュリティ] > [システム・セキュリティ] > [システムワイドセキュリティパラメータ]) で、[デフォルトセキュリティドメイン] ドロップダウンを使用して既定のドメインを選択します。
   c. [保存] をクリックします。

   このページの詳細は、"システム規模のセキュリティ・パラメータ" を参照してください。

注釈　複数のドメインを使用する場合も、各ユーザの名前は一意である必要があります。ユーザのタイプが異なっていても、一意でなければなりません。したがって、パスワード・ユーザである EndUser@example.com などのユーザを作成した場合、ユーザ EndUser@example.com として LDAP を介して InterSystems IRIS にログインすることはできません。InterSystems IRIS では、EndUser@example.com のアカウントを LDAP ユーザとして作成できないためです。

必要なログイン・ロールの設定

InterSystems IRIS のインスタンスが複数あり、LDAP 認証または OS ベースの認証と LDAP 承認を使用する場合、インスタンスに接続するユーザに必要なロールを各インスタンスで作成することを強くお勧めします。このメカニズムにより、ユーザは、十分な特権がないインスタンスにはアクセスできなくなります。このようにしないと、あるインスタンスでさまざまなロールを保持しているユーザが、意図していないインスタンスで同じロールを持つ可能性があります。

必要なログイン・ロールを設定する手順は以下のとおりです。

1. インスタンスごとに、必要なロールがまだ存在しない場合は、"ロールの作成" の手順に従ってロールを作成します。
2. インスタンスごとに、必要なロールを [**システムセキュリティ設定**] ページの [**このシステムに接続するのに必要なロール**] フィールド ([**システム管理**]→[**セキュリティ**]→[**システム・セキュリティ**]→[**システムワイドセキュリティパラメータ**]) で指定します。
3. 必要なロールの名前を含む名前の LDAP グループを追加します。グループの名前の形式は以下のとおりです。

   intersystems-Instance-instanceID**-Role-**rolename

   各要素の内容は以下のとおりです。

   - instanceID は、LDAP サーバにおけるインスタンスの一意の識別子です。
   - rolename は、接続に必要なロールの名前です。

注釈　ミラーリングを使用している場合など、特定の状況では、複数のインスタンス間で必要な 1 つのログイン・ロールを作成する方が望ましいこともあります。

例えば、2 つのシステム TEST と PRODUCTION があるとします。それらのシステムのセキュリティを個別に確保するには、**TEST** に **TESTACCESS** ロールを作成し、**PRODUCTION** に **PRODUCTIONACCESS** ロールを作成します。**TEST** では、[**このシステムに接続するのに必要なロール**] フィールドの値に **TESTACCESS** を設定します。**PRODUCTION** では、このフィールドの値に **PRODUCTIONACCESS** を設定します。その後で、**TEST** システムへのアクセスのみをユーザに許可する場合は、そのユーザに **TESTACCESS** ロールのみを割り当てて、**PRODUCTIONACCESS** ロールは割り当てないようにします。両方のシステムにアクセスできるユーザには、**PRODUCTIONACCESS** ロールと **TESTACCESS** ロールの両方を割り当てます。

サービスおよびアプリケーションでの LDAP の有効化

インスタンスで LDAP 認証を有効にしたら、インスタンスの関連サービスまたはアプリケーションで LDAP 認証を有効にします。

1. LDAP 認証がインスタンスで有効になっているため、LDAP 認証をサポートするサービスの [**サービス編集**] ページと Web アプリケーションの [**ウェブ・アプリケーション編集**] ページに [LDAP] チェック・ボックスが表示されます。
2. 必要に応じて、サービスおよびアプリケーションで LDAP 認証を有効にします。

LDAP 認証をサポートしているサービスは以下のとおりです。

- **%Service_Bindings**
- **%Service_CallIn**
- **%Service_ComPort**

- **%Service_Console**
- **%Service_Login**
- **%Service_Terminal**
- **%Service_Telnet**
- **%Service_WebGateway**

これらのサービスは、アクセス・モードによっていくつかのカテゴリに分類されます。

- ローカル・アクセス ー

  %Service_CallIn、**%Service_ComPort**、**%Service_Console**、**%Service_Login**、**%Service_Terminal**、**%Service_Telnet**

  ローカル接続で LDAP 認証を使用するには、サービスの LDAP 認証を有効にします。

- クライアント・サーバ・アクセス ー

  %Service_Bindings

  クライアント・サーバ接続で LDAP 認証を使用するには、サービスの LDAP 認証を有効にします。

- Web アクセス ー

  %Service_WebGateway

  指定したユーザが LDAP 認証を使用して Web アプリケーションにログインできるためには、関連の Web アプリケーションで LDAP を使用できるようにする必要があります。Web アプリケーションに認証メカニズムを追加する方法の詳細は、"アプリケーション" の "セキュリティの設定" を参照してください。サービスで LDAP を有効にすると、Web ゲートウェイ自体でも LDAP 認証を使用した認証が可能になります。

LDAP 認証後のインスタンスの状態

LDAP 認証を使用して最初に認証されるユーザは、“LDAP ユーザ” というタイプで **[ユーザ]** ページ (**[システム管理]→[セキュリティ]→[ユーザ]**) のユーザ・テーブルに表示されます。システム管理者が管理ポータル (またはその他の InterSystems IRIS ネイティブ機能) を使用して明示的にユーザを作成した場合、そのユーザのタイプは “InterSystems IRIS パスワード・ユーザ” になります。ユーザが LDAP 認証を使用してログインを試行し、認証が正常に行われる場合でも、そのユーザが既に (LDAP ユーザではなく) InterSystems IRIS ユーザとして存在することを InterSystems IRIS が検出すると、ログインは失敗します。

ポータルでの LDAP 構成の %Operator としての表示

管理ポータルに **%Operator** ロールまたは **%Admin_Operate:Use** 特権を持つユーザとしてログインしている場合は、インスタンスの LDAP 構成を表示できます (ただし、編集はできません)。

1. ポータルで、**[LDAP 構成]** ページ (**[システム処理]** > **[LDAP 構成]**) に移動します。
2. このページで、表示する構成の名前をクリックすると、その構成の **[LDAP 構成の表示]** が表示されます。

LDAP 構成を編集するには、**[セキュリティ LDAP 構成]** ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[LDAP 構成]**) に移動します。**%Admin_Secure:Use** 特権を持っている必要があります。

[セキュリティ LDAP 構成] ページ

ポータルの **[セキュリティ LDAP 構成]** ページ (**[システムオペレーション]→[LDAP 構成]**) には、インスタンスの LDAP 構成のリストが表示されます。構成名をクリックすると、そのプロパティが表示されます。インスタンスに対して Kerberos 認証が有効になっている場合、このページは **[セキュリティ LDAP/Kerberos 構成]** ページという名前になります (**[システム処理]** > **[LDAP/Kerberos 構成]**)。

LDAP 承認

LDAP を使用した認証の実行に加え、InterSystems IRIS では LDAP 承認がサポートされます。ロール、ルーチン、およびネームスペース定義の管理には、LDAP 属性ではなく LDAP グループを使用することをお勧めします。

LDAP 承認の構成の概要

承認に LDAP を使用するようにインターシステムズのサービスまたはアプリケーションを構成する手順は以下のとおりです。

1. LDAP 認証または OS ベースの認証を使用できるようにインスタンスを構成します。
2. LDAP 承認について、以下の手順を実行します。
   a. InterSystems IRIS インスタンスにおける LDAP 承認のグループを設計します。
   b. それらのグループを使用するように LDAP サーバを構成します。

LDAP グループを使用した承認の構成

- LDAP グループと InterSystems IRIS
- LDAP 承認グループ・モデル
  - 1 つのインスタンスを対象とする LDAP 承認グループ (単一インスタンス・グループ) の作成
  - 複数のインスタンスを対象とする LDAP 承認グループ (複数インスタンス・グループ) の作成 (ミラーリングの承認を含む)
  - ユニバーサル LDAP 承認グループの作成
- LDAP グループを使用した LDAP 承認に関するその他のトピック

LDAP グループと InterSystems IRIS

LDAP グループを使用すると、LDAP サーバを使用してユーザに特権を割り当てることができます。

- LDAP サーバのスキーマによって、グループの名前が指定されます。通常は、LDAP 管理者がこれらの名前を定義します。InterSystems IRIS では、後述する、事前定義された 3 つの名前構造のいずれかが使用されます。
- 各グループには、一意に識別できる識別名 (DN) があります。
- 各グループでは、InterSystems IRIS のロール、ルーチン、またはネームスペースへのアクセスを指定します。

InterSystems IRIS は、以下のインスタンスを対象として承認を提供する LDAP グループをサポートしています。

- 1 つのインスタンス
- 複数のインスタンス
- すべてのインスタンス

InterSystems IRIS で使用するグループを設定する手順は以下のとおりです。

1. 1 つのインスタンス、複数のインスタンス、すべてのインスタンスのいずれを対象とするグループを使用するかを決定します。
2. 適切な命名規則に準拠する名前で 1 つ以上のグループを作成します。各グループでは、ユーザのロール、既定のネームスペース、または既定のルーチンを指定します。ユーザは複数のロールを持つことができるので、ロールを指定する複数のグループに属することもできます。

   注釈　LDAP サーバ上でこれらのグループを定義する際、ディストリビューション・グループとしてではなく、セキュリティ・グループとして作成する必要があります。

3. LDAP ユーザを構成して、どのユーザがどのグループに属するかを指定します。そのためには、各ユーザの LDAP アカウントについて、1 つ以上のロール、既定のネームスペース、および既定のルーチンを指定する複数のグループにユーザを割り当てる必要があります。これにより、ログイン後に各ユーザに付与されるロール、ユーザの既定のネームスペース、およびユーザの既定のルーチンが決まります。
4. LDAP サーバで指定されたすべてのロールの定義が含まれるように、ローカル InterSystems IRIS インスタンスを構成します。

LDAP 承認グループ・モデル

InterSystems IRIS では、LDAP を使用した 3 種類のグループ承認がサポートされています。

- 1 つのインスタンスを対象とする LDAP 承認グループ (単一インスタンス・グループ) の作成
- 複数のインスタンスを対象とする LDAP 承認グループ (複数インスタンス・グループ) の作成 (ミラーリングを含む)
- ユニバーサル LDAP 承認グループの作成

1 つのインスタンスを対象とする LDAP 承認グループ (単一インスタンス・グループ) の作成

InterSystems IRIS では、1 つのインスタンスのみを対象として承認を提供する LDAP グループを作成できます。このような各グループは、単一インスタンス・グループと呼ばれます。この種類の承認グループを作成する手順は以下のとおりです。

1. InterSystems IRIS インスタンスで、LDAP パラメータ **[承認インスタンス ID]** の値を確認または変更します。既定では、その値は NodeName_InstanceName です。NodeName は InterSystems IRIS インスタンスが実行されているマシン、InstanceName はそのインスタンスの名前です。

   パラメータの値を手動で設定するには、以下の手順を実行します。

   a. 管理ポータルで、**[セキュリティ LDAP 構成]** ページ (**[管理ポータル]** > **[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[LDAP 構成]**) に移動します。

   b. そのページで、編集する構成の名前をクリックして選択します。

   c. 表示される、構成を編集するためのページで、**[ロール、ルーチン、ネームスペースで LDAP グループを使用する]** を選択します。

   d. 次に、**[承認インスタンス ID]** フィールドにパラメータの値を入力し、**[保存]** をクリックします。

2. LDAP サーバで、`Instance` キーワードに続けて **[承認インスタンス ID]** の値を使用して、必要なインターシステムズの構造に準拠する名前でロール、ネームスペース、およびルーチンのグループを定義します。これらの文字列は大文字と小文字を区別しません。これらのグループ名の形式は以下のとおりです。

   `intersystems-Instance-`AuthorizationInstanceIDValue`-Role-`RoleName

   `intersystems-Instance-`AuthorizationInstanceIDValue`-Routine-`RoutineName

   `intersystems-Instance-`AuthorizationInstanceIDValue`-Namespace-`NamespaceName

   各要素の内容は以下のとおりです。

   - AuthorizationInstanceIDValue は、[承認インスタンス ID] フィールドに指定された値です。
   - RoleName、RoutineName、および NamespaceName はそれぞれ、ロール、既定のルーチン、または既定のネームスペースの名前です。

     注釈　ユーザは、任意の数のロールを持つことができます。通常、システムへのアクセスには少なくとも 1 つのロールが必要です。ユーザは、1 つの既定のルーチンおよび 1 つの既定のネームスペースのみを持つことができます。ただし、これらは必須ではないため、ユーザが既定のルーチンや既定のネームスペースを持たなくてもかまいません。

- RoleName には、複数のロールを “^” で区切って含めることができます。例えば、“%All^Admin^Application4” には、“%All” ロール、“Admin” ロール、および “Application4” ロールが含まれます。

3. InterSystems IRIS インスタンスで、各グループに関連するロールを構成します。

例えば、Node1 というマシン上にある Test というインスタンスでアプリケーションを実行しているとします。以下の 3 つのユーザ・カテゴリを設定します。

- アプリケーション・ユーザ ー アプリケーションのみを実行できます。
- 管理ユーザ ー さまざまな管理ツールとアプリケーションを実行できます。
- スーパーユーザ ー フル・アクセス権を持ちます。

この承認モデルを設定するには、LDAP サーバで以下のグループを作成します。

```
intersystems-Instance-Node1_Test-Role-Administrator
intersystems-Instance-Node1_Test-Role-LocalApplication
intersystems-Instance-Node1_Test-Role-%All
intersystems-Instance-Node1_Test-Routine-LocalApplication
intersystems-Instance-Node1_Test-Routine-%SS
intersystems-Instance-Node1_Test-Routine-%pmode
intersystems-Instance-Node1_Test-Namespace-%SYS
intersystems-Instance-Node1_Test-Namespace-USER
```

次に、各ユーザ・カテゴリに対応するロールを作成します。

- 管理者
- LocalApplication

注釈　**%All** ロールは既に存在するため、作成する必要はありません。

最後に、3 つのユーザ・カテゴリを作成します。

- アプリケーション・ユーザ ー アプリケーション LocalApplication のみを実行できます。以下の LDAP グループに割り当てられます。
  - `intersystems-Instance-Node1_Test-Role-LocalApplication`
  - `intersystems-Instance-Node1_Test-Routine-LocalApplication`
  - `intersystems-Instance-Node1_Test-Namespace-USER`
- 管理ユーザ ー さまざまな管理ツールとアプリケーションを実行できます。以下の LDAP グループに割り当てられます。
  - `intersystems-Instance-Node1_Test-Role-LocalApplication`
  - `intersystems-Instance-Node1_Test1-Role-Administrator`
  - `intersystems-Instance-Node1_Test-Routine-%SS`
  - `intersystems-Instance-Node1_Test-Namepace-%SYS`
- スーパーユーザ ー **%All** アクセス権を持ちます。以下の LDAP グループに割り当てられます。
  - `intersystems-Instance-Node1_Test-Role-%All`
  - `intersystems-Instance-Node1_Test-Namespace-%SYS`
  - `intersystems-Instance-Node1_Test-Routine-%pmode`

複数のインスタンスを対象とする LDAP 承認グループ (複数インスタンス・グループ) の作成

InterSystems IRIS では、複数のインスタンスを対象として承認を提供する LDAP グループを作成できます。このような各グループは、複数インスタンス・グループと呼ばれます。この種類の承認グループを作成する手順は以下のとおりです。

1. 個々のインスタンスがグループ間でどのように情報を共有するかを決定します。これにより、各インスタンスのグループとユーザがアクセスできる情報が決まります。
2. グループ内の各インスタンスについて、LDAP パラメータ **[承認グループ ID]** の値をグループ内の他のインスタンスと同じになるように変更します。

   パラメータの値を手動で設定するには、以下の手順を実行します。

   a. 管理ポータルで、**[セキュリティ LDAP 構成]** ページ (**[管理ポータル]** > **[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[LDAP 構成]**) に移動します。

   b. そのページで、編集する構成の名前をクリックして選択します。

   c. 表示される、構成を編集するためのページで、**[ロール、ルーチン、ネームスペースで LDAP グループを使用する]** を選択します。

   d. 次に、**[承認グループ ID]** フィールドにパラメータの値を入力し、**[保存]** をクリックします。

3. LDAP サーバで、`Group` キーワードに続けて **[承認グループ ID]** の値を使用して、必要なインターシステムズの構造に準拠するロール、ネームスペース、およびルーチンのグループを設定します。これらの文字列は大文字と小文字を区別しません。これらのグループ名の形式は以下のとおりです。

   `intersystems-Group-`AuthorizationGroupIDValue`-Role-`RoleName

   `intersystems-Group-`AuthorizationGroupIDValue`-Routine-`RoutineName

   `intersystems-Group-`AuthorizationGroupIDValue`-Namespace-`NamespaceName

   各要素の内容は以下のとおりです。

   - AuthorizationGroupIDValue は、[承認グループ ID] フィールドに指定された値です。
   - RoleName、RoutineName、および NamespaceName はそれぞれ、ロール、既定のルーチン、または既定のネームスペースの名前です。

     注釈　ユーザは、任意の数のロールを持つことができます。通常、システムへのアクセスには少なくとも 1 つのロールが必要です。ユーザは、1 つの既定のルーチンおよび 1 つの既定のネームスペースのみを持つことができます。ただし、これらは必須ではないため、ユーザが既定のルーチンや既定のネームスペースを持たなくてもかまいません。

   - RoleName には、複数のロールを“^”で区切って含めることができます。例えば、“%All^Admin^Application4”には、“%All”ロール、“Admin”ロール、および“Application4”ロールが含まれます。

4. それらを使用するすべてのインスタンスで、必要なロールを構成します。

例えば、5 台のデータベース・サーバに接続された 7 台の ECP アプリケーション・サーバがあるとします。データベース・サーバのうちの 2 台はフェイルオーバー・ペアで、それ以外の 3 台は非同期レポート・メンバです。これらのサーバ (アプリケーション・サーバとデータベース・サーバの両方) はすべて、SALES アプリケーションを実行します。アプリケーションのエンド・ユーザに必要な特権のセットは限定されますが、その管理ユーザはより多くの特権を必要とします。したがって、以下の 3 つのユーザ・カテゴリを設定します。

- アプリケーション・ユーザ ー アプリケーションのみを実行できます。
- アプリケーション・サーバ管理者 ー アプリケーションを実行できます。アプリケーション・サーバへのフル・アクセス権を持ちますが、データベース・サーバにはアクセスできません。

- データベース管理者 – アプリケーション・サーバへのフル・アクセス権とデータベース・サーバへの管理アクセス権を持ちます。

これらの要件をサポートするように LDAP 承認を構成する手順は以下のとおりです。

- アプリケーション・サーバの [**承認グループ ID**] を SALESAPP に設定します。
- データベース・サーバの [**承認グループ ID**] を SALESDB に設定します。

LDAP サーバで、グループを以下のように定義します。

```
intersystems-Group-SALESAPP-Role-%All
intersystems-Group-SALESAPP-Role-LocalApplication
intersystems-Group-SALESAPP-Routine-LocalApplication
intersystems-Group-SALESAPP-Routine-%pmode
intersystems-Group-SALESAPP-Namespace-USER
intersystems-Group-SALESAPP-Namespace-%SYS
intersystems-Group-SALESDB-Role-Administrator
intersystems-Group-SALESDB-Routine-INTEGRIT
intersystems-Group-SALESDB-Namespace-%SYS
```

次に、各ユーザ・カテゴリに対応するロールを作成します。

- 管理者
- LocalApplication

注釈　**%All** ロールは既に存在するため、作成する必要はありません。

最後に、3 つのユーザ・カテゴリを作成します。

- アプリケーション・ユーザ – アプリケーション LocalApplication のみを実行できます。以下の LDAP グループに割り当てられます。
  - `intersystems-Group-SALESAPP-Role-LocalApplication`
  - `intersystems-Group-SALESAPP-Routine-LocalApplication`
  - `intersystems-Group-SALESAPP-Namespace-USER`
- アプリケーション・サーバ管理者 – アプリケーションを実行できます。アプリケーション・サーバへのフル・アクセス権を持ちますが、データベース・サーバにはアクセスできません。以下の LDAP グループに割り当てられます。
  - `intersystems-Group-SALESAPP-Role-LocalApplication`
  - `intersystems-Group-SALESAPP-Namespace-USER`
  - `intersystems-Group-SALESAPP-Role-%All`
  - `intersystems-Group-SALESAPP-Routine-%pmode`
- データベース管理者 – アプリケーション・サーバへのフル・アクセス権とデータベース・サーバへの管理アクセス権を持ちます。以下の LDAP グループに割り当てられます。
  - `intersystems-Group-SALESAPP-Role-%All`
  - `intersystems-Group-SALESAPP-Routine-%pmode`
  - `intersystems-Group-SALESAPP-Namespace-%SYS`
  - `intersystems-Group-SALESDB-Role-Administrator`
  - `intersystems-Group-SALESDB-Routine-INTEGRIT`

- intersystems-Group-SALESDB-Namespace-%SYS

この時点で、完全に機能する承認モデルが存在しますが、データベース・サーバへの (**%All** を持つ) スーパーユーザ・アクセスが含まれていません。このようなアクセスを追加するには、ユーザを作成して以下の新しいグループに追加します。

intersystems-Group-SALESDB-Role-%All

ミラーリングを含む LDAP 承認グループの構成

LDAP とミラーリングを使用する場合は、複数インスタンス LDAP グループを使用して承認を構成することをお勧めします。必要な複数インスタンス・グループを作成し、それらのグループを使用するようにすべてのメンバ (非同期メンバを含む) のすべてのユーザを構成します。

前述の例で定義したグループ構造に基づく、以下の例を検討してみましょう。さらに、以下のように仮定します。

- フェイルオーバー・ペアである SALESDBMIR というミラーと 3 つのレポート非同期メンバがあります。
- **%All** を持つユーザを作成しますが、フェイルオーバー・ペアのみでこのアクセス権を持ちます。

このミラーの承認を構成する手順は以下のとおりです。

1. フェイルオーバー・ペアへのフル・アクセス権を提供するために、以下のグループを作成します。

   intersystems-Group-SALESDBMIRFAILOVER-Role-%All

2. 非同期メンバへのフル・アクセス権を提供するために、以下のグループを作成します。

   intersystems-Group-SALESDBMIRASYNC-Role-%All

3. フェイルオーバー・ペアの各メンバの LDAP パラメータ [**承認インスタンス ID**] を SALESDBMIRFAILOVER に設定します。

   重要　災害復旧 (DR) 非同期メンバはフェイルオーバー・メンバに昇格することがあるため、DR 非同期メンバの [**承認インスタンス ID**] も SALESDBMIRFAILOVER に設定する必要があります。

4. ミラーの非同期メンバの LDAP パラメータ [**承認グループ ID**] を SALESDBMIRASYNC に設定します。
5. 次に、アプリケーション・サーバへの **%All** アクセス権、ミラーリングされていないデータベース・サーバへの管理アクセス権、およびフェイルオーバー・ペアのみへの **%All** アクセス権を持つミラー管理者を作成します。これらのユーザは、以下の LDAP グループに割り当てられます。
   - intersystems-Group-SALESAPP-Role-%All
   - intersystems-Group-SALESAPP-Routine-%pmode
   - intersystems-Group-SALESAPP-Namespace-%SYS
   - intersystems-Group-SALESDB-Role-Administrator
   - intersystems-Group-SALESDB-Routine-INTEGRIT
   - intersystems-Group-SALESDB-Namespace-%SYS
   - intersystems-Group-SALESDBMIRFAILOVER-Role-%All
6. 最後に、すべてのメンバ (アプリケーション・サーバ、データベース・サーバ、フェイルオーバー・ペア、および非同期メンバ) への **%All** アクセス権を持つフル管理者を作成します。これらのユーザは、以下の LDAP グループに割り当てられます。

- intersystems-Group-SALESAPP-Role-%All
- intersystems-Group-SALESDB-Role-%All
- intersystems-Group-SALESDBMIRFAILOVER-Role-%All
- intersystems-Group-SALESDBMIRASYNC-Role-%All

ユニバーサル LDAP 承認グループの作成

InterSystems IRIS では、1 つの LDAP サーバを使用するすべてのインスタンスを対象として承認を提供する LDAP グループを作成できます。これらは、ユニバーサル承認グループと呼ばれます。この種類の承認グループを作成する手順は以下のとおりです。

1. 現在のインスタンスでユニバーサル承認グループの使用を有効にします。
   a. 管理ポータルで、**[セキュリティ LDAP 構成]** ページ (**[管理ポータル]** > **[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[LDAP 構成]**) に移動します。
   b. そのページで、編集する構成の名前をクリックして選択すると、その構成を編集するためのページが表示されます。
   c. 構成を編集するためのページで、**[ロール、ルーチン、ネームスペースで LDAP グループを使用する]** を選択します。
   d. **[ユニバーサルグループ承認を許可する]** を選択します。
   e. **[保存]** をクリックします。
2. LDAP サーバで、必要なインターシステムズの構造に準拠するロール、ネームスペース、およびルーチンのグループを設定します。これらの文字列は大文字と小文字を区別しません。これらのグループ名の形式は以下のとおりです。

   `intersystems-Role-`RoleName

   `intersystems-Routine-`RoutineName

   `intersystems-Namespace-`NamespaceName

   RoleName、RoutineName、および NamespaceName はそれぞれ、ロール、既定のルーチン、または既定のネームスペースの名前です。RoleName には、複数のロールを“^”で区切って含めることができます。例えば、“%All^Admin^Application4”には、“%All”ロール、“Admin”ロール、および“Application4”ロールが含まれます。

   注釈　ユーザは、任意の数のロールを持つことができます。通常、システムへのアクセスには少なくとも 1 つのロールが必要です。ユーザは、1 つの既定のルーチンおよび 1 つの既定のネームスペースのみを持つことができます。ただし、これらは必須ではないため、ユーザが既定のルーチンや既定のネームスペースを持たなくてもかまいません。
3. LDAP サーバを使用するすべてのインスタンスで、必要なロールを構成します。

例えば、LocalApplication というアプリケーションがあり、LDAP サーバを使用するすべての InterSystems IRIS インスタンスで、そのアプリケーションに対するさまざまなレベルのアクセス権をユーザに付与するとします。以下の LDAP グループを定義します。

```
intersystems-Role-%All
intersystems-Role-Administrator
intersystems-Role-LocalApplication
intersystems-Routine-%SS
intersystems-Routine-LocalApplication
intersystems-Namespace-USER
intersystems-Namespace-%SYS
```

次に、各ユーザ・カテゴリに対応するロールを作成します。

- Admin
- LocalApplication

注釈　**%All** ロールは既に存在するため、作成する必要はありません。

最後に、3 つのユーザ・カテゴリを作成します。

- アプリケーション・ユーザ ー すべてのサーバ上のアプリケーションにアクセスできます。以下の LDAP グループに割り当てられます。
  - `intersystems-Role-LocalApplication`
  - `intersystems-Routine-LocalApplication`
  - `intersystems-Namespace-USER`
- 管理者 ー すべてのサーバへの管理アクセス権を持ちます。以下の LDAP グループに割り当てられます。
  - `intersystems-Role-Administrator`
  - `intersystems-Routine-%SS`
  - `intersystems-Namespace-%SYS`
- スーパーユーザ ー すべてのサーバへのフル・アクセス権を持ちます。以下の LDAP グループに割り当てられます。
  - `intersystems-Role-%All`

LDAP グループを使用した LDAP 承認に関するその他のトピック

トピックは以下のとおりです。

- LDAP グループ定義の構造
- LDAP グループ名の構成
- さまざまな種類のグループの混合
- 入れ子になったグループ
- LDAP グループによって InterSystems IRIS へのアクセスが規制される仕組み

LDAP グループ定義の構造

通常、グループ定義の内容は以下のとおりです。

- グループ名
- グループの組織単位の宣言 (`OU=Groups`)
- ドメイン・コンポーネント (DC) の宣言 (`DC=example,DC=com` など)
- 必要なその他の情報

例えば、以下のようなグループ定義が考えられます。

```
CN=intersystems-Role-Administrator,OU=Groups,DC=intersystems,DC=com
CN=intersystems-Group-MyGroup-Namespace-USER,OU=Groups,DC=intersystems,DC=com
CN=intersystems-Instance-MyNode:MyInstance-Routine-INTEGRIT,OU=Groups,DC=intersystems,DC=com
```

LDAP グループ名の構成

InterSystems IRIS では、LDAP グループ名をさらに細かく構成できます。以下のセクションでは、既定の構成、構成可能なプロパティ、およびプロパティの変更手順について説明します。

既定のグループ名の構成

既定では、LDAP グループ名では以下の構文を使用します。

`intersystems-Role-`RoleName

`intersystems-Routine-`RoutineName

`intersystems-Namespace-`NamespaceName

`intersystems-Group-`GroupName`-Role-`RoleName

`intersystems-Group-`GroupName`-Routine-`RoutineName

`intersystems-Group-`GroupName`-Namespace-`NamespaceName

`intersystems-Instance-`InstanceName`-Role-`RoleName

`intersystems-Instance-`InstanceName`-Routine-`RoutineName

`intersystems-Instance-`InstanceName`-Namespace-`NamespaceName

グループ名のプロパティ

グループ名は、以下の構成可能なプロパティで構成されます。

- OrganizationID ー 既定値は `intersystems` です。グループ名の `intersystems` セグメントをユーザ定義文字列または空文字列に置き換えます。例えば、`OrgABC` に設定した場合、グループ名は以下のようになります。

  `OrgABC-Role-`RoleName

  `OrgABC-Group-`GroupName`-Routine-`RoutineName

  `OrgABC-Instance`InstanceName`-Namespace-`NamespaceName

  空文字列に設定した場合、グループ名は以下のようになります。

  `Role-`RoleName

  `Group-`GroupName`-Routine-`RoutineName

  `Instance-`InstanceName`-Namespace-`NamespaceName

- DelimiterID ー 既定値は hyphen (-) です。これは、グループ名に含まれるセグメントを区切る区切り文字です。例えば、アンダースコア (_) に設定した場合、グループ名は以下のようになります。

  `intersystems_Role_`RoleName

  `intersystems_Group_`GroupName`_Routine_`RoutineName

  `intersystems_Instance_`InstanceName`_Namespace_`NamespaceName

- GroupID ー 既定値は `Group` です。例えば、`SystemGrouping` に設定した場合、グループ名は以下のようになります。

  `intersystems-SystemGrouping-`GroupName`-Role-`RoleName

  `intersystems-SystemGrouping-`GroupName`-Routine-`RoutineName

intersystems-SystemGrouping-GroupName-Namespace-NamespaceName

- InstanceID ー 既定値は Instance です。例えば、SystemInstance に設定した場合、グループ名は以下のようになります。

  intersystems-SystemInstance-InstanceName-Role-RoleName

  intersystems-SystemInstance-InstanceName-Routine-RoutineName

  intersystems-SystemInstance-InstanceName-Namespace-NamespaceName

- RoleID ー 既定値は Role です。例えば、SystemRole に設定した場合、グループ名は以下のようになります。

  intersystems-SystemRole-RoleName

  intersystems-Group-GroupName-SystemRole-RoleName

  intersystems-Instance-InstanceName-SystemRole-RoleName

- NamespaceID ー 既定値は Namespace です。例えば、SystemNamespace に設定した場合、グループ名は以下のようになります。

  intersystems-SystemNamespace-NamespaceName

  intersystems-Group-GroupName-SystemNamespace-NamespaceName

  intersystems-Instance-InstanceName-SystemNamespace-NamespaceName

- RoutineID ー 既定値は Routine です。例えば、SystemRoutine に設定した場合、グループ名は以下のようになります。

  intersystems-SystemRoutine-RoutineName

  intersystems-Group-GroupName-SystemRoutine-RoutineName

  intersystems-Instance-InstanceName-SystemRoutine-RoutineName

プロパティの変更手順

これらのプロパティを変更するには、以下の手順を実行します。

1. 管理ポータルで、[セキュリティ LDAP 構成] ページ ([管理ポータル] > [システム管理] > [セキュリティ] > [システム・セキュリティ] > [LDAP 構成]) に移動します。
2. 構成を編集するには、構成の名前をクリックします。
3. このページでは、OrganizationID プロパティを編集できます。[詳細設定] をクリックし、残りのプロパティを表示して編集します。
4. ページの上部の [保存] をクリックして、変更を保存します。

さまざまな種類のグループの混合

単一インスタンス・ロールまたは複数インスタンス・ロールと組み合わせてユニバーサル・グループを使用できます。

例えば、以下のように仮定します。

- 複数のインスタンスに、あるアプリケーションがあります。
- ユニバーサル・グループを使用しています。
- すべてのインスタンスでアプリケーションを実行できますが、いずれのマシンでも管理者としてアプリケーションを使用することはできない UserOne というユーザが存在します。

UserOne が以下の操作を行えるようにします。

- 引き続き、すべてのインスタンスでアプリケーションを実行できる
- さらに、Test という特定のマシン上の APPTEST という特定のインスタンスでアプリケーションを管理できる

以下はその方法です。

1. Test マシン上の APPTEST インスタンスの承認インスタンス ID を Test:APPTEST に設定します。
2. LDAP サーバで以下のグループを作成します。

   ```
   intersystems-Instance-Test_APPTEST-Role-Administrator
   ```

3. LDAP サーバでこのグループを UserOne に割り当てます。
4. Test マシン上の APPTEST インスタンスで管理者ロールを作成し、管理アクセス権を付与します。

他の方法で承認グループを組み合わせて使用することもできます。例えば、LDAP サーバに対して認証されるすべてのインスタンスで UserTwo が **%All** 許可を持っている場合、Server10 というマシン上の SECRET というインスタンスで UserTwo に排他管理許可を付与できます。そのためには、**[ユニバーサル・グループ・アクセスを許可]** を無効にし、`intersystems-Instance-Server10_SECRET-Role-Administrator` をそのユーザに割り当てるプロセスを実行します。

入れ子になったグループ

Active Directory LDAP サーバで、LDAP グループは入れ子になったグループと呼ばれるものをサポートします。入れ子になったグループは、親グループのメンバであるグループです。つまり、入れ子になったグループのメンバであるユーザはすべて、暗黙的に親グループのメンバでもあります。例えば、ABC と DEF と呼ばれる 2 つの LDAP グループが定義されているとします。ABC を、DEF 内の入れ子になったグループにすることができます。つまり、ユーザが ABC のメンバである場合、明示的に DEF グループに割り当てなくても、このグループのメンバになります。

ユーザの入れ子になったグループを検索すると、LDAP サーバ上でセキュリティ・グループとして定義されたグループのみが返されます。入れ子になったグループを使用する場合、InterSystems IRIS システムでロールとして使用するグループはセキュリティ・グループとして作成するようにしてください。

注釈　入れ子になったグループを使用しないシステムでは、セキュリティ・グループとディストリビューション・グループの両方が返されます。

LDAP グループによって InterSystems IRIS へのアクセスが規制される仕組み

ユーザは、LDAP グループを通じて、既定のネームスペースおよび既定のルーチンと共にロールを受け取ります。ユーザに付与されたロールに、インスタンスの必要なアクセス・ポイントに対する十分な特権がない場合、ユーザはそのインスタンスへのアクセスを拒否されます。例えば、既定のルーチンを使用する十分な特権がユーザにない場合、そのユーザはアクセスを拒否されます。

さらに、以下のルールが適用されます。

- ユーザがロールのグループに割り当てられていても、ユーザがログインしているインスタンスでそのロールが定義されていない場合、ユーザはそのインスタンスではそのロールを持ちません。
- ユーザが既定のルーチンのグループに割り当てられていても、ユーザがログインしているインスタンスでそのルーチンが定義されていない場合、ユーザはそのインスタンスには接続できません。
- ユーザが既定のネームスペースのグループに割り当てられていても、ユーザがログインしているインスタンスでそのネームスペースが定義されていない場合、ユーザはそのインスタンスには接続できません。

オペレーティング・システム・ベースの認証と併用する LDAP 承認の構成

トピックは以下のとおりです。

- オペレーティング・システム LDAP 認証
- InterSystems IRIS インスタンスでの OS/LDAP の有効化
- %Service_Console および %Service_Terminal サービスでの OS/LDAP の有効化
- 単一ドメインおよび複数ドメインによる OS/LDAP
- 入力要求を簡略化するための複数ドメインによる OS/LDAP の構成

オペレーティング・システム LDAP 認証

InterSystems IRIS では、オペレーティング・システム・ベースの認証をサポートするように構成することで、LDAP 経由の承認を実行できます。これは、オペレーティング・システム LDAP 承認または OS/LDAP と呼ばれます。これにより、ユーザは、オペレーティング・システム・ログインからの認証情報を使用して InterSystems IRIS に対して認証を行った後、LDAP サーバから承認情報を取得することができます。オペレーティング・システム LDAP 承認は、Windows のコンソールと、UNIX®、Linux、および macOS のターミナルで使用できます。

OS/LDAP を構成する手順は以下のとおりです。

1. InterSystems IRIS インスタンスで OS ベースの認証と LDAP 承認を有効化します。
2. 標準の LDAP 認証と同様に、インスタンスにログインするために必要なロールを設定します。
3. %Service_Console および %Service_Terminal サービスで OS/LDAP を有効化します。
4. 承認を構成します。これは、LDAP 認証に追加する場合と同じ方法で行います。"InterSystems IRIS の LDAP 承認の構成" を参照してください。
5. 複数のドメインを使用する場合は、オプションで入力要求を簡略化するように OS/LDAP を構成します。

InterSystems IRIS インスタンスでの OS/LDAP の有効化

OS/LDAP を使用するには、まず、インスタンスで OS/LDAP を有効にします。

1. 管理ポータルのホーム・ページで、**[認証/Web セッション・オプション]** ページ (**[システム管理]** > **[セキュリティ]** > **[システム・セキュリティ]** > **[認証/Web セッション・オプション]**) に移動します。
2. **[認証/Web セッション・オプション]** ページで、**[オペレーティング・システム LDAP 認証を許可]** を選択します。
3. **[保存]** をクリックすると、変更内容が適用されます。

%Service_Console および %Service_Terminal サービスでの OS/LDAP の有効化

インスタンスの関連サービスまたはアプリケーションで OS/LDAP を有効にする手順は以下のとおりです。

1. インスタンスで LDAP 認証が有効になっている場合、OS/LDAP をサポートするサービスである **%Service_Console** および **%Service_Terminal** の **[サービス編集]** ページに **[オペレーティング・システム LDAP 承認]** チェック・ボックスが表示されます。
2. 必要に応じて、それらのサービスで LDAP 認証を有効にします。

単一ドメインおよび複数ドメインによる OS/LDAP

OS/LDAP では、単一ドメインまたは複数ドメインの使用がサポートされています。

1 つのドメインのみをサポートするように InterSystems IRIS が構成されている場合、以下のように動作します。

1. ユーザは、最初のログイン時にユーザ名とパスワードの入力を求められます。
2. それ以降のログインについては、オペレーティング・システムによってユーザが既に認証されているため、入力を求められることはありません。

複数のドメインをサポートするように InterSystems IRIS が構成されている場合、以下のように動作します。

1. ユーザは、最初のログイン時にユーザ名とパスワードの入力を求められます。
2. それ以降のログインについては、既定では、オペレーティング・システムによってユーザ名とパスワードの入力が求められます。この入力要求が行われないように InterSystems IRIS を構成できます。次のセクションを参照してください。

入力要求を簡略化するための複数ドメインによる OS/LDAP の構成

OS/LDAP と複数のドメインを使用する場合、入力要求を簡略化するようにインスタンスを構成できます。既定では、ユーザはログインのたびにユーザ名とパスワードの入力を求められます。ユーザが最初にログインするときにのみユーザ名/パスワードの入力が求められ、それ以降の接続は入力要求なしで認証されるように InterSystems IRIS を構成できます。

このように動作するように InterSystems IRIS を構成する手順は以下のとおりです。

1. それぞれのユーザについて、ユーザが認証されるドメインの値を持つ環境変数 ISC_LDAP_CONFIGURATION を作成します。
2. ユーザが認証される各ドメインについて、以下の手順を実行します。
   a. LDAP 構成があることを確認するか、LDAP 構成を作成します。
   b. その LDAP 構成について、**[ISC_LDAP_CONFIGURATION 環境変数を許可]** チェック・ボックスにチェックを付けます。これにより、この環境変数を使用できるようになります。

LDAP 属性を使用した承認の構成

LDAP 承認には、LDAP グループを使用することをお勧めします。ただし、インターシステムズでは、LDAP 属性を使用した承認もサポートしています。承認情報を格納するために LDAP スキーマで使用できる OID が 3 つ登録されています。それぞれには、以下のような固有の目的があります。

- intersystems-Namespace ー ユーザの既定のネームスペースの名前 (OID 1.2.840.113556.1.8000.2448.2.1)。
- intersystems-Routine ー ユーザの既定のルーチンの名前 (OID 1.2.840.113556.1.8000.2448.2.2)。
- intersystems-Roles ー ユーザのログイン・ロールの名前 (OID 1.2.840.113556.1.8000.2448.2.3)。

これらの属性を使用するには、LDAP サーバで以下の手順を実行します。

1. 属性を有効にして使用できるようにします。そのためには、LDAP スキーマの objectClass フィールドで、その値のリストに `intersystemsAccount` の値を付加してこのフィールドの値を変更します(`intersystemsAccount` には 1.2.840.113556.1.8000.2448.1.1 という LDAP OID が設定されています)。
2. このフィールドを (必要なだけ) スキーマに追加します。
3. LDAP データベースのエントリにそれらの値を生成します。

注釈　登録済みの LDAP スキーマ名を使用する必要はありません。実際、使用している LDAP スキーマにある既存の属性を使用できます。

例えば、UNIX® LDAP サーバでは、InterSystems IRIS での LDAP 認証の使用に関するスキーマを定義するために、以下の定義に示されている内容を使用します。

```
# Attribute Type Definitions

attributetype ( 1.2.840.113556.1.8000.2448.2.1 NAME 'intersystems-Namespace'
       DESC 'InterSystems Namespace'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.5 SINGLE-VALUE )

attributetype ( 1.2.840.113556.1.8000.2448.2.2 NAME 'intersystems-Routine'
        DESC 'InterSystems Routine'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} SINGLE-VALUE )

attributetype ( 1.2.840.113556.1.8000.2448.2.3 NAME 'intersystems-Roles'
        DESC 'InterSystems Roles'
```

```
        EQUALITY caseIgnoreMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# Object Class Definitions

objectclass ( 1.2.840.113556.1.8000.2448.1.1
        NAME 'intersystemsAccount'
        SUP top
        AUXILIARY
        DESC 'Abstraction of an account with InterSystems attributes'
        MAY (
                intersystems-Routine $
                intersystems-Namespace $
                intersystems-Roles
        )
)
```

このコンテンツは以下の 2 か所に配置します。

- /etc/openldap/schema/ ディレクトリの intersystems.schema ファイルに配置します。
- 他のコンテンツと共に、/etc/openldap/slapd.conf ファイルに組み込みます。

LDAP に関するその他のトピック

安全なアウトバウンド LDAP 接続の作成

このドキュメントでは主に、InterSystems IRIS に接続する際の認証と承認での LDAP の使用について説明していますが、InterSystems IRIS から LDAP サーバに接続することもできます。LDAP サーバへの安全なアウトバウンド接続を確立するために、InterSystems IRIS は TLS をサポートしています。このトピックの詳細は、Init メソッドのコンテンツにある %SYS.LDAP のクラス・ドキュメントを参照してください。

LDAP API の使用法

%SYS.LDAP クラスでは、プログラムによって LDAP がサポートされます。

UNIX® の認証で InterSystems IRIS LDAP API を使用していて、詳細なデバッグ情報が必要な場合、OpenLDAP パッケージの一部である ldapsearch プログラムを使用できます。認証に関する問題を修正したら、構成のテスト・ツールを使用して、接続が機能していることを確認できます。ldapsearch プログラムは、LDAP 接続の他の問題のデバッグにも役に立つ場合があります。

さまざまな LDAP アクションの動作

このセクションでは、LDAP 認証および承認に関連する特定のプロセス中に行われる処理について説明します。

- LDAP で認証および承認が実行される仕組み
- LDAP データベース内でターゲット・ユーザが検索される仕組み
- インスタンスで LDAP アカウントの条件に基づいてローカル・アカウントがチェックおよび削除される仕組み

LDAP で認証および承認が実行される仕組み

LDAP 認証を使用する InterSystems IRIS のインスタンスに対して認証を試行するプロセスは以下のようになります。

1. ユーザは、ユーザ名とパスワードの入力を要求されます。認証を試行するこのユーザはターゲット・ユーザと呼ばれます。
2. InterSystems IRIS は、[**検索に使用するLDAPユーザ名**] と [**LDAPユーザ名パスワード**] に指定された値を使用して、LDAP サーバへの接続を確立します。InterSystems IRIS が情報を取得できるように LDAP データベースを検索する特権を持つこのユーザは検索ユーザと呼ばれます。
3. 接続が確立されると、次に、[**LDAPユニーク検索属性**] を使用して LDAP データベース内でターゲット・ユーザが検索されます。

4. LDAP データベース内でターゲット・ユーザが見つかると、そのユーザに関連付けられた属性 (ユーザのロール、ネームスペース、ルーチンなど) が取得されます。
5. その後で、InterSystems IRIS は LDAP データベースに対してユーザの認証を試行します。このとき、手順 1 で入力したユーザ名とパスワードが使用されます。
6. 認証が成功すると、LDAP サーバで (グループ割り当てまたは属性を使用して) 承認が行われます。ユーザはその後、ロールおよび公開されている利用可能なリソースに関連付けられた特権に基づいて InterSystems IRIS と対話できます。管理ポータルに表示されるユーザのプロパティは読み取り専用で、InterSystems IRIS 内で編集することはできません。

LDAP データベース内でターゲット・ユーザが検索される仕組み

InterSystems IRIS が検索ユーザとして LDAP サーバへの接続を確立したら、次にターゲット・ユーザに関する情報を取得します。そのために、InterSystems IRIS は、ログイン時に入力されたユーザ名を LDAP Unique search attribute に対する LDAP データベース内の値と照合して確認します。この属性の名前は多くの場合、Active Directory LDAP サーバでは “sAMAccountName”、OpenLDAP サーバでは “uid” です。

InterSystems IRIS によりユーザが検索されると、属性情報が取得されます。InterSystems IRIS により、InterSystems IRIS LDAP 構成フィールド (″LDAP 構成の作成または変更″ で説明しています) で指定された各属性に関する情報が取得され、各属性に関連付けられたすべての値が取得されます。InterSystems IRIS では、InterSystems IRIS LDAP 構成フィールド内でユーザに指定されたすべての属性に関連付けられているすべての値が取得されることに注意してください。サブセットのみを取得するように構成することはできません。

インスタンスで LDAP アカウントの条件に基づいてローカル・アカウントがチェックおよび削除される仕組み

アカウントが以下のいずれかの条件を満たす場合、InterSystems IRIS ではローカル・インスタンスのユーザ・アカウントが削除されます。

- LDAP アカウントが既に存在しない
- LDAP アカウントが無効である
- Active Directory のみで、LDAP アカウントにパスワードの変更を要求するフラグが設定されている
- Active Directory のみで、LDAP アカウントが失効している

InterSystems IRIS は、以下の状況でこれらの条件をチェックし、アカウントを削除します。

- ユーザが InterSystems IRIS インスタンスにログインしようとすると、インスタンスによってユーザの LDAP アカウントがチェックされます。LDAP アカウントについて、記載された条件のいずれかを満たす場合、InterSystems IRIS はローカル・ユーザ・アカウントを削除します。
- セキュリティ・スキャン・タスクの結果として。InterSystems IRIS にはこのタスクが用意されています。タスクを実行して、ローカル・ユーザ・アカウントに関連付けられた LDAP アカウントについて、これらのいずれかの条件を満たすかどうかを確認します。いずれかの条件を満たす場合、InterSystems IRIS はローカル・ユーザ・アカウントを削除します。

OAuth 2.0 および OpenID Connect

OAuth 2.0 および OpenID Connect の概要

ここでは、OAuth 2.0 承認フレームワークと OpenID Connect の概要を説明します。別のページでは、OAuth 2.0 および OpenID Connect に対する InterSystems IRIS® のサポートについて説明します。

基本情報

OAuth 2.0 承認フレームワークを使用すると、サードパーティのアプリケーション (一般にクライアントと呼ばれる) による HTTP サービス (リソース) への限定的なアクセスを可能にすることができます。このアクセス制限により、クライアントが取得できる情報や使用できるサービスが限定されます。承認サーバは承認のやりとりを調整するか、アクセス権を直接付与します。OpenID Connect は、このフレームワークを拡張して認証を追加します。

ロール

OAuth 2.0 フレームワークでは、次の 4 つのロールが定義されています。

- リソース所有者 ー 通常はユーザです。
- リソース・サーバ ー 保護されているデータやサービスをホストするサーバです。
- クライアント ー リソース・サーバへの限定的なアクセスを要求するアプリケーションです。これは、クライアント・サーバ・アプリケーションの場合もあれば、サーバを持たないアプリケーション (JavaScript アプリケーションやモバイル・アプリケーションなど) の場合もあります。
- 承認サーバ ー リソース・サーバへのクライアントのアクセスを可能にするアクセス・トークンの発行を担うサーバです。このサーバは承認サーバと同じアプリケーションにすることも、異なるアプリケーションにすることもできます。

クライアント、リソース・サーバ、承認サーバは事前の準備によって相互に認識します。承認サーバは、通信可能なクライアント・サーバとリソース・サーバを指定したクライアントのレジストリを保持しています。クライアントが登録されると、承認サーバはクライアント ID とクライアント秘密鍵を生成します。クライアント秘密鍵は秘密にしておく必要があります。クライアントが承認サーバと通信する方法によっては、クライアント・サーバでクライアント秘密鍵が必要になることもあります。その場合は、クライアント秘密鍵をクライアント・サーバへ安全に伝える必要があります。JavaScript アプリケーションなどの一部のシナリオでは、クライアントはクライアント秘密鍵を保護することができません。このようなシナリオでは、クライアントはクライアント秘密鍵を必要としない方法で承認サーバと通信しなければなりません。

アクセス・トークン

アクセス・トークンには、ユーザやクライアントの ID に関する情報が、有効期限、予期される発行者名、予期される対象者、スコープなどのメタデータと共に格納されます。

通常、アクセス・トークンの目的は、HTTP 経由で提供されるリソース・サーバの特定のデータやサービスにクライアントがアクセスできるようにすることです。全体的な流れでは、クライアント・アプリケーションが承認サーバにアクセス・トークンを要求します。クライアントは、受け取ったアクセス・トークンをリソース・サーバへの HTTP 要求で使用します。リソース・サーバは、受け取った要求に有効なアクセス・トークンが含まれている場合にのみ、要求された情報を返します。

アクセス・トークンは取り消すこともできます (承認サーバがこの機能をサポートしている場合)。

アクセス・トークンの形態

InterSystems IRIS は、次の 2 種類のアクセス・トークンをサポートしています。

- JSON Web Token (JWT)。JWT は JSON オブジェクトです。JWT はデジタル署名か暗号化またはその両方を行うことができます。

  JWT の 1 種が ID トークンであり、OpenID Connect に固有のトークンです。

  JWT は署名か暗号化またはその両方を行うことができます。

- opaque アクセス・トークン (別名 : 参照トークン)。この形態のアクセス・トークンは、他の場所、具体的には承認サーバに格納されているトークンの識別子です。この識別子は長いランダムな文字列であり、これには推測を困難にするという意図が込められています。

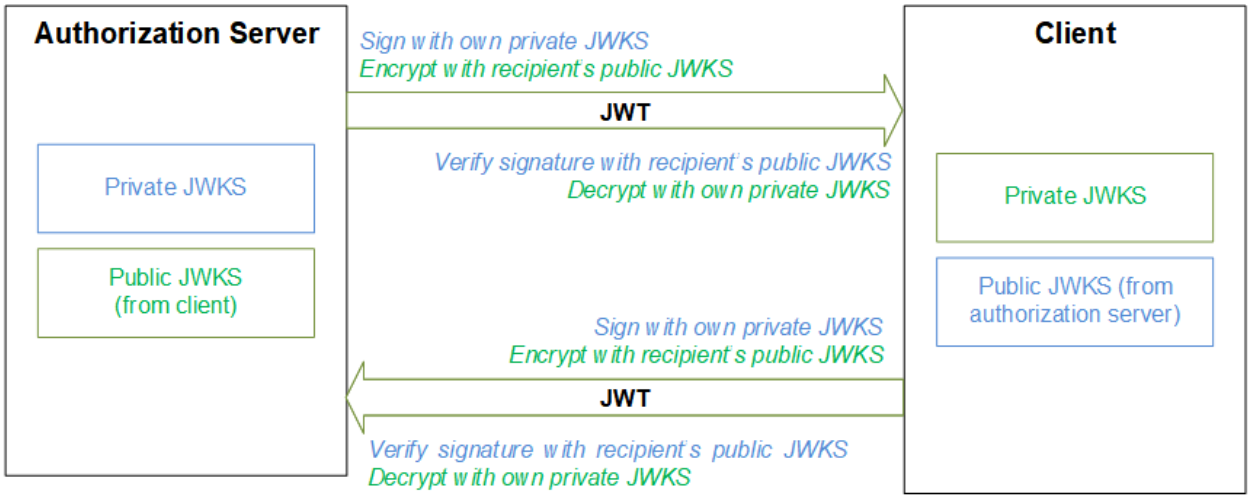
クレーム

アクセス・トークンには、ユーザまたはクライアントの ID の伝達や、トークンの有効期限、予期される発行者名、予期される対象者、スコープなどのメタデータの伝達を行う一連のクレームが含まれています。OpenID Connect Core の仕様ではクレームの標準セットが定義されていますが、その他のクレームも使用できます。

JWT および JWKS

上記のように、JWT は署名か暗号化またはその両方を行うことができます。ほとんど場合、OAuth 2.0 フレームワークの参加者は、この目的で JWKS (JSON Web Key Set) のペアを使用します。JWKS の任意のペアにおいて、一方の JWKS は秘密鍵で、(アルゴリズムあたり) 必要なすべての秘密鍵だけでなく、対称鍵として使用するクライアント秘密鍵も含んでいます。この JWKS は共有されません。他方の JWKS には、対応する公開鍵が含まれていて、公開されて利用できます。

それぞれの参加者は、秘密の JWKS を持っていて、対応する公開の JWKS を他の参加者に提供します。秘密の JWKS の所有者は、その JWKS を使用して、送信 JWT に署名し、受信 JWT を解読します。他の参加者は、以下の図に示すように、対応する公開の JWKS を使用して送信 JWT を暗号化し、受信 JWT の署名を確認します。



付与タイプと付与フロー

OAuth 2.0 フレームワークでは、承認サーバが承認の要求を処理する方法が付与タイプによって指定されます。クライアントは、承認サーバへの最初の要求に付与タイプを指定します。OAuth 2.0 仕様には 4 つの付与タイプと、追加のタイプを定義する拡張メカニズムが記載されています。通常、それぞれの付与タイプは異なる全体フローに相当します。

4 つの付与タイプは以下のとおりです。

- 承認コード – この付与タイプは対応するサーバを持つクライアント・アプリケーションでのみ使用することができます。この付与タイプでは、ユーザがユーザ名とパスワードを入力するログイン・ページが承認サーバによって表示されます。これらの情報はクライアントと共有されることはありません。ユーザ名とパスワードが有効なユーザと一致すると (要求の他の要素が適切な場合)、承認サーバは最初に承認コードを発行し、これをクライアントに返します。クライアントは、この承認コードを使用してアクセス・トークンを取得します。

  承認コードの要求はブラウザに表示され、その応答も同様です。一方、アクセス・トークンの要求はサーバ間のやりとりであり、その応答も同様です。したがって、アクセス・トークンはブラウザに表示されることはありません。

  Proof Key for Code Exchange (PKCE) は、横取りした承認コードで悪意のあるアクターがアクセス・トークンを取得できないようにする、承認コード・フローの拡張です。PKCE では、クライアントの承認コードの要求に、追加のシークレット値が含められます。承認サーバは承認コードの発行時に、このシークレットを保存します。以降のクライアント

の承認コードをアクセス・トークンに交換する要求には、元のシークレットを含める必要があります。承認コードを横取りしたアクターはこのシークレットを知らないため、アクターがアクセス・トークンを取得するのを防ぐことができます。

- 暗黙 – 前述の付与タイプと同様に、承認サーバはログイン・ページを表示しますが、クライアントはユーザの資格情報にアクセスすることはできません。ただし、この暗黙付与タイプでは、クライアントがアクセス・トークンを直接要求して受け取ります。この付与タイプは、JavaScript クライアントやモバイル・アプリケーションなどの純正のクライアント・アプリケーションで役立ちます。
- リソース所有者のパスワード資格情報 – この付与タイプでは、クライアントがユーザ名とパスワードを入力するようにユーザに要求し、それにより得られた資格情報を使用して承認サーバからアクセス・トークンを取得します。信頼できるアプリケーションだけがこの付与タイプに適しています。
- クライアント資格情報付与タイプ – この付与タイプではユーザ・コンテキストは存在せず、クライアント・アプリケーションは人の介入なしに処理を行います。クライアントは、クライアント ID とクライアント秘密鍵を使用して承認サーバからアクセス・トークンを取得します。

RFC 7523 では、追加の付与タイプである JWT 承認について説明しています。この付与タイプでは、JSON Web トークン (JWT) ベアラー・トークンを使用して、OAuth 2.0 アクセス・トークンを要求し、クライアントを認証します。InterSystems IRIS は、OAuth 2.0 仕様の 4 つに加えて、この付与タイプをサポートしています。

通常、OAuth 2.0 フレームワークでは、すべての HTTP 要求が SSL/TLS で保護されています。

さらに、クライアントが承認サーバに要求を送信するときは、その要求を認証する必要があります。OAuth 2.0 仕様に、クライアントが要求を認証する方法が記述されています。

スコープ

承認サーバは、クライアントが `scope` 要求パラメータを使用してアクセス要求のスコープを指定するのを許可します。次に、承認サーバは `scope` 応答パラメータを使用して、発行したアクセス・トークンのスコープをクライアントに通知します。

OpenID Connect は OAuth 2.0 承認プロセスの拡張機能です。認証を要求するために、クライアントは承認サーバへの要求に `openid` スコープ値を組み込みます。承認サーバは、認証に関する情報を ID トークンと呼ばれる JWT で返します。ID トークンには、OpenID Connect Core 仕様に記載されている特定のクレーム・セットが含まれています。

承認サーバのエンドポイント

承認サーバは、さまざまな種類の要求を処理できる、以下の URL またはエンドポイントの一部またはすべてを提供します。

エンドポイント	目的
承認エンドポイント	承認コードを返します (承認コード付与タイプにのみ適用)。
トークン・エンドポイント	アクセス・トークンを返します。
Userinfo エンドポイント	認証されたユーザに関するクレームを収めた JSON オブジェクトを返します (OpenID Connect にのみ適用)。
トークン・イントロスペクション・エンドポイント	アクセス・トークンを調べて決定したクレームを収めた JSON オブジェクトを返します。
トークン取り消しエンドポイント	トークンを取り消します。

OAuth 2.0 クライアントとしての InterSystems IRIS Web アプリケーションの使用法

このページでは、OAuth 2.0 フレームワークを使用するクライアント・アプリケーションとして InterSystems IRIS® の Web アプリケーションを使用する方法について説明します。主に、InterSystems IRIS Web アプリケーションが Web サーバ/

クライアント・アプリケーションのクライアントとして機能し、承認コード付与タイプを使用するシナリオについて説明します。"OAuth 2.0 クライアントのバリエーション" も参照してください。

注釈　OAuth 2.0 クライアントが Active Directory フェデレーション・サービス (ADFS) 承認サーバと通信する際、クライアント・コードで承認エンドポイントに特別なキーと値のペアを付加する必要があります。詳細は、"メソッドの詳細" の "`GetAuthorizationCodeEndpoint()`" の説明を参照してください。

InterSystems IRIS クライアントの前提条件

このページに記載されているタスクを開始する前に、以下のものが利用可能であることを確認します。

- OAuth 2.0 承認サーバ。後で、このサーバの具体的な詳細情報が必要になります。以下に示す詳細項目の一部は、InterSystems IRIS 内でクライアントを構成するときに適用されます。
  - 承認サーバの場所 (発行者エンドポイント)
  - 承認エンドポイントの場所
  - トークン・エンドポイントの場所
  - Userinfo エンドポイントの場所 (サポートされている場合。OpenID Connect Core を参照)
  - トークン・イントロスペクション・エンドポイントの場所 (サポートされている場合。RFC 7662 を参照)
  - トークン取り消しエンドポイントの場所 (サポートされている場合。RFC 7009 を参照)
  - 承認サーバによるダイナミック登録のサポートの有無

  以下に示すその他の詳細情報は、クライアント・コードの作成時に必要になります。
  - このサーバがサポートする付与タイプ
  - このサーバがサポートするスコープ。例えば、OpenID Connect Core で定義されている特別なスコープである `openid` および `profile` をサーバがサポートすることもあれば、サポートしないこともあります。
  - このサーバに対して行われる要求に関するその他の要件
- 承認サーバがダイナミック・クライアント登録をサポートしない場合、InterSystems IRIS アプリケーションを OAuth 2.0 承認サーバのクライアントとして登録する必要があります。また、このクライアントのクライアント ID とクライアント秘密鍵が必要です。詳細は、承認サーバの実装環境によって異なります。(サーバがダイナミック登録をサポートする場合、このページの説明に従って構成するときにクライアントを登録できます。)

構成要件

OAuth 2.0 クライアントとして InterSystems IRIS Web アプリケーションを使用するには、次の構成タスクを実行します。

- InterSystems IRIS にサービスを提供している Web サーバでは、その Web サーバが SSL を使用するように構成します。SSL を使用するように Web サーバを構成する方法は、このドキュメントの対象外であるため、ここでは取り上げません。
- クライアントが使用する InterSystems IRIS の SSL 構成を作成します。

  これはクライアント SSL 構成です。証明書は不要です。この構成は Web サーバへの接続に使用されます。この接続を通じて、クライアントは、承認サーバと通信してアクセス・トークンを取得し、Userinfo エンドポイントの呼び出し、イントロスペクション・エンドポイントの呼び出しなどを行います。

  SSL 構成の作成の詳細は、インターシステムズの "TLS ガイド" を参照してください。

  それぞれの SSL 構成には一意の名前が付いています。参照用に、このドキュメントではこの SSL 構成を `sslconfig` と呼びますが、一意の名前を任意に付けることができます。

- クライアントの OAuth 2.0 構成項目を作成します。そのためには、サブセクションの説明に従って最初にサーバ記述を作成し、次にクライアント構成を作成します。

  両項目の必要なオプションを管理ポータルで見つけるために、**[システム管理]→[セキュリティ]→[OAuth 2.0]→[クライアント構成]** を選択します。このページには、(承認サーバとして使用されているマシン以外の) クライアント・マシンで OAuth 2.0 構成を作成するときに必要なオプションが表示されます。

  クライアント・マシンでは、**[システム管理]→[セキュリティ]→[OAuth 2.0]→[サーバ構成]** のメニューを使用しないでください。

このタスクを実行するには、**%Admin_OAuth2_Client** リソースの USE 権限を持つユーザとしてログインする必要があります。

サーバ記述の作成 (Discovery の使用)

1. 管理ポータルで **[システム管理]→[セキュリティ]→[OAuth 2.0]→[クライアント構成]** を選択します。

   このインスタンスで利用可能なサーバ記述を示すページが表示されます。どの行でも、**[発行者エンドポイント]** 列はサーバ記述の発行者エンドポイントを示します。**[クライアント数]** 列は、指定されたサーバ記述に関連付けられたクライアント構成の数を示します。最後の列では **[クライアント構成]** リンクにより、関連付けられているクライアント構成の作成、表示、編集、削除を行うことができます。

2. **[サーバ構成の作成]** を選択します。

   管理ポータルに、サーバ記述の詳細を入力できる新たなページが表示されます。

3. 以下の詳細情報を指定します。

   - **[発行者エンドポイント]** (必須) – 承認サーバの識別に使用するエンドポイントの URL を入力します。
   - **[SSL/TLS 構成]** (必須) – ダイナミック・クライアント登録要求時に使用する SSL/TLS 構成を選択します。
   - **[登録アクセス・トークン]** – オプションで、ダイナミック・クライアント登録要求を承認するためのベアラー・トークンとして使用する初期登録アクセス・トークンを入力します。

4. **[検出と保存]** を選択します。

   次に InterSystems IRIS は、指定された承認サーバと通信し、サーバ記述に必要な情報を取得して、情報を保存します。

   管理ポータルにサーバ記述の一覧が再度表示されます。

サーバ記述の手動による作成 (Discovery なし)

(Discovery を使用せずに) 手動でサーバ記述を作成するには、まずサーバ記述ページを表示して (上記の手順 1 および 2)、次に **[手動]** を選択します。これによって、以下のようにページに多数のオプション・セットが表示されます。

- **[発行者エンドポイント]** (必須) – 承認サーバの識別に使用するエンドポイントの URL を入力します。
- **[承認エンドポイント]** (必須) – 承認サーバに承認コードを要求するときに使用するエンドポイント URL を入力します。
- **[トークン・エンドポイント]** (必須) – 承認サーバにアクセス・トークンを要求するときに使用するエンドポイント URL を入力します。
- **[Userinfo エンドポイント]** (必須) – 承認サーバのアクセス・トークンを使用して承認の Userinfo 要求を行うときに使用するエンドポイント URL を入力します。
- **[トークン・イントロスペクション・エンドポイント]** – `client_id` と `client_secret` を使用して承認のトークン・イントロスペクション要求を行うときに使用するエンドポイント URL を入力します。RFC 7662 を参照してください。
- **[トークン取り消しエンドポイント]** – `client_id` と `client_secret` を使用して承認のトークン取り消し要求を行うときに使用するエンドポイント URL を入力します。RFC 7009 を参照してください。

- [JSON Web Token (JWT) の設定] – JWT のシグニチャの検証および解読を承認サーバからクライアントが行うために使用する公開鍵のソースを指定します。

  既定では、承認サーバは JWKS (JSON Web Key Set) のペアを生成します。一方の JWKS は秘密鍵で、(アルゴリズムあたり) 必要なすべての秘密鍵だけでなく、対称鍵として使用するクライアント秘密鍵も含んでいます。この JWKS は共有されません。他方の JWKS には、対応する公開鍵が含まれていて、公開されて利用できます。また、サーバ記述を作成するプロセスでも、JWT のシグニチャの検証と暗号化に使用するために、承認サーバからクライアントに公開の JWKS がコピーされます。

  - [URL から JWKS] – 公開の JWKS を指す URL を指定した後、InterSystems IRIS に JWKS をロードします。
  - [ファイルから JWKS] – 公開の JWKS を含むファイルを選択し、そのファイルを InterSystems IRIS にロードします。
  - [X509 証明書] – 詳細は、"証明書と JWT (JSON Web Token)" の "OAuth 2.0 承認サーバの証明書の使用" を参照してください。

  これらのオプションのいずれかにアクセスするには、まず [ダイナミック登録以外のソース] を選択します。

これらの値を指定し、[保存] を選択します。

クライアントの構成および動的な登録

ここでは、クライアントの構成を作成し、クライアントを動的に登録する方法について説明します。

1. 管理ポータルで [システム管理]→[セキュリティ]→[OAuth 2.0]→[クライアント構成] を選択します。

   管理ポータルにサーバ記述の一覧が表示されます。

2. このクライアント構成を関連付けるサーバ記述の行で [クライアント構成] リンクをクリックします。

   管理ポータルに、サーバ記述に関連付けられているクライアント構成の一覧が表示されます。これは、最初は空のリストです。

3. [クライアント構成の作成] をクリックします。

   管理ポータルに、詳細を入力できる新たなページが表示されます。

4. [一般] タブで、以下の詳細を指定します。

   - [アプリケーション名] – アプリケーションの短い名前を指定します。
   - [クライアント名] – エンド・ユーザに表示するクライアント名を指定します。
   - [説明] – アプリケーションの説明を入力します (オプション)。
   - [有効] – このアプリケーションが使用されないようにする場合は、このチェック・ボックスのチェックを外すこともできます。
   - [クライアント・タイプ] – 以下のいずれかを選択します。
     - [機密] – RFC 6749 に従って、クライアントを機密クライアントに指定します。

       このページでは、クライアントが承認コード付与タイプを使用するシナリオについて主に説明します。このシナリオでは、[クライアント・タイプ] に [機密] を指定します。その他の付与タイプは、"バリエーション" を参照してください。
     - [パブリック] – RFC 6749 に従って、クライアントをパブリック・クライアントに指定します。
     - [リソース・サーバ] – クライアントを専用のリソース・サーバに指定します。
   - [SSL/TLS 構成] – クライアントで使用するために作成した SSL 構成を選択します (例: `sslconfig`)。

- **[応答を受信するために承認サーバに対して指定されるクライアント URL]** – InterSystems IRIS OAuth 2.0 クライアントで必要とされる内部宛先の URL を指定します。この宛先でアクセス・トークンが保存され、ブラウザは元のクライアント・アプリケーションへリダイレクトされます。

  この URL を指定するには、次のオプションの値を入力します。

  - **[ホスト名]** – 承認サーバのホスト名または IP アドレスを指定します。
  - **[ポート]** – Web ゲートウェイ構成の変更に対応するために必要な場合は、この値を指定します。
  - **[接頭語]** – Web ゲートウェイ構成の変更に対応するために必要な場合は、この値を指定します。

  指定した URL は以下の形式をとります。

  ```
  https://hostname:port/prefix/csp/sys/oauth2/OAuth2.Response.cls
  ```

  [ポート] の指定を省略すると、コロンが省略されます。同様に、**[接頭語]** の指定を省略すると、`hostname:port` と `csp` の間のスラッシュが 1 つだけになります。(さらに、**[TLS/SSL を使用する]** オプションのチェックを外すと、URL は `https` ではなく `http` で始まります。)

- **[TLS/SSL を使用する]** – リダイレクト・ページを開くときに TLS/SSL を使用しない正当な理由がある場合を除き、このオプションを選択します。
- **[フロントチャネルログアウト URL]** – 必要に応じて HTTP ベースのフロントチャネル・ログアウト URL を指定します。サーバはこの URL を登録し、それを使用してクライアント上のユーザをログアウトします。フロントチャネル・ログアウトをサポートしないクライアントを作成するには、この URL を空白のままにします。このフィールドの上のボックスには指定した URL が表示され、それに 'IRISLogout=end' が追記されます。

  注釈　InterSystems IRIS クライアントでフロントチャネル・ログアウトがサポートされるようにするには、クライアント・アプリケーションの **[セッション Cookie のスコープ]** を **[　]** に設定します。アプリケーションの設定の構成の詳細は、"アプリケーションの作成および編集" を参照してください。

- **[必要な付与タイプ]** – クライアントで限定使用する OAuth 2.0 付与タイプを指定します。
- **[認証タイプ]** – 承認サーバへの HTTP 要求で使用する認証タイプを選択します (RFC 6749 または OpenID Connect Core のセクション 9 で規定)。**[なし]**、**[基本]**、**[フォームエンコードされた本文]**、**[クライアント秘密鍵 JWT]**、または **[秘密鍵 JWT]** のいずれかを選択します。
- **[認証サーバがログアウトURLを呼び出す際に iss および sid クエリパラメータが必要]** – 承認サーバからフロントチャネル・ログアウト URL が呼び出されるときにクエリ・パラメータとして iss (発行者) と sid (セッション ID) を必要とするには、このオプションを選択します。
- **[認証署名アルゴリズム]** – トークン・エンドポイントでこのクライアントを認証するために使用される JWT の署名に必要なアルゴリズムを選択します ([認証タイプ] が **[クライアント秘密鍵 JWT]** または **[秘密鍵 JWT]** である場合)。オプションを選択しない場合、OpenID プロバイダおよび Relying Party でサポートされる任意のアルゴリズムが使用されます。

5. **[クライアント情報]** タブで、以下の詳細を指定します。
   - **[ロゴ URL]** – クライアント・アプリケーションのロゴの URL です。
   - **[クライアント・ホームページ URL]** – クライアント・アプリケーションのホーム・ページの URL です。
   - **[ポリシー URL]** – クライアント・アプリケーションのポリシー・ドキュメントの URL です。
   - **[サービス条件の URL]** – クライアント・アプリケーションのサービス条件ドキュメントの URL です。
   - **[デフォルトのスコープ]** – アクセス・トークン要求の既定のスコープを空白で区切ったリストで指定します。この既定値は、承認サーバで許可されているスコープと一致している必要があります。

- **[連絡先電子メール]** – クライアント・アプリケーションの責任者への連絡に使用する適切な電子メール・アドレスをコンマで区切ったリストです。
- **[既定の最長経過時間]** – 既定の最長認証経過時間 (秒単位) を指定します。このオプションを指定した場合、最長認証経過時間に達したとき、エンド・ユーザをアクティブに再認証する必要があります。max_age 要求パラメータは、この既定値をオーバーライドします。このオプションを省略した場合に、既定で最長認証経過時間は設定されません。

6. **[JWT 設定]** タブで、以下の詳細を指定します。
   - **[X509 資格情報から JWT 設定を作成]** – 証明書に関連付けた秘密鍵を署名と暗号化で使用する場合は、このオプションを選択します。この場合は、"証明書と JWT (JSON Web Token)" の "OAuth 2.0 クライアントの証明書の使用" も参照してください。

     注釈　インターシステムズでは、**[X509 資格情報から JWT 設定を作成]** オプションが使用されることはまれで、次に説明する既定の動作が代わりに使用されると考えています。

     このオプションのチェックを外したままにすると、システムは JWKS (JSON Web Key Set) のペアを生成します。一方の JWKS は秘密鍵で、(アルゴリズムあたり) 必要なすべての秘密鍵だけでなく、対称鍵として使用するクライアント秘密鍵も含んでいます。この JWKS は共有されません。他方の JWKS には、対応する公開鍵が含まれていて、公開されて利用できます。ダイナミック登録プロセスでは、公開の JWKS を承認サーバにもコピーし、承認サーバがこのクライアントから JWT の暗号化およびシグニチャの検証を実行できるようにします。
   - **[署名アルゴリズム]** – 署名済み JWT の作成に使用する署名アルゴリズムを選択します。JWT を署名しない場合は空白のままにします。
   - **[暗号化アルゴリズム]** – 暗号化された JWT の作成に使用する暗号化アルゴリズムを選択します。JWT を暗号化しない場合は空白のままにします。値を選択する場合は、**[キー・アルゴリズム]** も指定する必要があります。
   - **[キー・アルゴリズム]** – 暗号化された JWT の作成に使用するキー管理アルゴリズムを選択します。JWT を暗号化しない場合は空白のままにします。

7. 承認サーバがダイナミック登録をサポートする場合は、入力したすべてのデータを再確認した後、**[ダイナミック登録と保存]** をクリックします。次に、InterSystems IRIS は承認サーバを接続し、クライアントを登録して、クライアント ID とクライアント秘密鍵を取得します。

   承認サーバがダイナミック登録をサポートしない場合については、この後のサブセクションを参照してください。

クライアントの構成 (ダイナミック登録なし)

承認サーバがダイナミック登録をサポートしない場合は、上記の最後の手順に代わって以下の手順を実行します。

1. **[クライアント資格情報]** タブを選択し、次の詳細を指定します。
   - **[クライアント ID]** – 承認サーバが提供したクライアント ID を入力します。
   - **[クライアント秘密鍵]** – 承認サーバが提供したクライアント秘密鍵を入力します。**[クライアント・タイプ]** が **[機密]** の場合は、この値が必須となります。

     このページでは、クライアントが承認コード付与タイプを使用するシナリオについて主に説明します。このシナリオでは、**[クライアント秘密鍵]** の値を指定します。その他の付与タイプは、"バリエーション" を参照してください。

   [クライアント ID 発行日時]、**[クライアント秘密鍵有効期限]**、および **[登録クライアント URI]** の値を入力しないでください。
2. **[保存]** を選択します。

コード要件の概要

注釈　このセクションでは、トークンの要求時にクライアントが承認コード付与タイプを使用するときに必要とされるコードについて説明します。その他の付与タイプは、"バリエーション" を参照してください。

InterSystems IRIS Web アプリケーションが OAuth 2.0 クライアントとして機能するには、この Web アプリケーションで以下のようなロジックを使用する必要があります。

1. アクセス・トークンと ID トークン (必要な場合) を取得します。"トークンの取得" を参照してください。
2. アクセス・トークンと (必要に応じて) ID トークンを検証し、要求されたリソースの使用に必要な権限をユーザが保有しているかどうかを確認します。"トークンの検証" を参照してください。
3. 権限が適切な場合は、"HTTP 要求へのアクセス・トークンの追加" の説明に従ってリソース・サーバを呼び出します。

以降のセクションでこれらの手順に関する情報を提供します。

トークンの取得

注釈　このセクションでは、トークンの要求時にクライアントが承認コード付与タイプを使用するときに必要とされるコードに関する情報を提供します。既定では、この承認コード付与タイプには、Proof Key for Code Exchange (PKCE) 拡張が含まれます。その他の付与タイプおよび PKCE なしの承認コードについては、"バリエーション" を参照してください。

トークンを取得するには、以下のようなトークン取得の手順を使用します。サブセクションでは、ここで使用するメソッドを詳しく説明します。

1. %SYS.OAuth2.AccessToken クラスの IsAuthorized() メソッドを呼び出します。その際、最初にアクセス・トークンの適切なスコープ (1 つ以上) を決定する必要があります。

   以下に例を示します。

   ObjectScript

   ```
   set myscopes="openid profile scope1 scope2"
   set isAuth=##class(%SYS.OAuth2.AccessToken).IsAuthorized("myclient",,myscopes,
                  .accessToken,.idtoken,.responseProperties,.error)
   ```

   このメソッドは、アクセス・トークンが既にローカルに保存されているかどうかを確認します。

2. error 引数がエラーを返したかどうかを確認し、返した場合はそのエラーを適切に処理します。この引数にエラーが含まれる場合、関数 $ISOBJECT() は 1 を返します。それ以外の場合、$ISOBJECT() は 0 を返します。

   ObjectScript

   ```
       if $isobject(error) {
     //error handling here
   }
   ```

3. IsAuthorized() が 1 を返す場合は "トークンの検証" へ進みます。
4. それ以外の場合は、%SYS.OAuth2.Authorization クラスの GetAuthorizationCodeEndpoint() メソッドを呼び出します。その際、以下の情報を必要とします。
   - アクセス・トークンを返した後に承認サーバがリダイレクトされる完全な URL。これは、クライアントのリダイレクト・ページです (元のページと同じにすることも、別のページにすることもできます)。
   - 要求のスコープ (1 つ以上)。
   - 要求に含まれるパラメータ。例えば、場合によっては `claims` パラメータを渡す必要があります。

以下に例を示します。

ObjectScript

```
set scope="openid profile scope1 scope2"
set redirect="https://localhost/csp/openid/SampleClientResult.csp"

set url=##class(%SYS.OAuth2.Authorization).GetAuthorizationCodeEndpoint("myclient",
      scope,redirect,.properties,.isAuthorized,.sc)
if $$$ISERR(sc) {
  //error handling here
}
```

このメソッドは、InterSystems IRIS OAuth 2.0 クライアントが必要とする内部宛先の完全な URL (クエリ・パラメータを含む) を返します。

このメソッドに対する Proof Key for Code Exchange (PKCE) の既定の動作を変更するには、"メソッドの詳細" で `properties` 引数の詳細を参照してください。

5. GetAuthorizationCodeEndpoint() が返した URL を開くオプション (ボタンなど) を提供することで、ユーザは要求を承認することができます。

   InterSystems IRIS は、ユーザには表示されないこの内部 URL で承認コードを取得し、それをアクセス・トークンに交換し、クライアントのリダイレクト・ページにブラウザをリダイレクトします。

メソッドの詳細

このサブセクションでは、前のサブセクションで使用したメソッドについて詳しく説明します。

IsAuthorized()

場所:このメソッドはクラス %SYS.OAuth2.AccessToken にあります。

```
ClassMethod IsAuthorized(applicationName As %String,
                         sessionId As %String,
                         scope As %String = "",
                         Output accessToken As %String,
                         Output IDToken As %String,
                         Output responseProperties,
                         Output error As %OAuth2.Error) As %Boolean
```

このクライアントおよびセッションのアクセス・トークンがローカルに保存され、さらにそのアクセス・トークンが scope 引数で提供されるすべてのスコープを承認する場合、このメソッドは 1 を返します。(このメソッドは IRISSYS データベースでアクセス・トークンを探します。これらのトークンは、有効期限が切れると自動的に削除されます。)

それ以外の場合、このメソッドは 0 を返します。

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。
- sessionId はセッション ID を指定します。既定のセッション (%session.SessionId) をオーバーライドする場合のみこれを指定します。
- scope は、スペースで区切られたスコープのリストです。例えば、`"openid profile scope1 scope2"` のような形式をとります。

  `openid` および `profile` は、OpenID Connect Core で定義されている特別なスコープです。
- 出力として返される accessToken はアクセス・トークンです (存在する場合)。
- 出力として返される IDToken は ID トークンです (存在する場合)。(これは、OpenId Connect を使用している場合にのみ適用されます。具体的には、要求でスコープ `openid` が使用された場合です。)ID トークンは JWT です。

- 出力として返される responseProperties は、応答のパラメータを含む多次元配列です。この配列の構造は以下のとおりです。

配列ノード	配列値
responseProperties(parametername)、ここで parametername はパラメータの名前です (token_type、expires_in など)。	指定されたパラメータの値。

- 出力として返される error は、空の文字列であるか (エラーがない場合)、エラー情報を含む %OAuth2.Error のインスタンスです (エラーがある場合)。

  %OAuth2.Error は、Error、ErrorDescription、ErrorUri という 3 つの文字列プロパティを持ちます。

GetAuthorizationCodeEndpoint()

場所:このメソッドはクラス %SYS.OAuth2.Authorization にあります。

```
ClassMethod GetAuthorizationCodeEndpoint(applicationName As %String,
                                         scope As %String,
                                         redirectURL As %String,
                                         ByRef properties As %String,
                                         Output isAuthorized As %Boolean,
                                         Output sc As %Status,
                                         responseMode As %String
                                         sessionId As %String = "") As %String
```

このメソッドは、InterSystems IRIS が承認コードを要求するときに使用するローカルな内部ページの URL を、必要なすべてのクエリ・パラメータと共に返します。(このページはユーザには表示されません。)

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。
- scope は、スペースで区切られた、アクセスが要求されているスコープのリストです。例えば、"scope1 scope2 scope3" のような形式をとります。

  既定値は、指定された applicationName のクライアント構成によって決まります。
- redirectURL は、クライアントのリダイレクト・ページの完全な URL です。クライアントにアクセス・トークンを返した承認サーバは、このページにブラウザをリダイレクトします。
- 参照によって渡される properties は、要求に追加されるパラメータを含む多次元配列です。この配列は以下の構造を持つ必要があります。

配列ノード	配列値
properties (parametername)、ここで parametername はパラメータの名前です。	指定されたパラメータの値。この値は、スカラ値、ダイナミック・オブジェクトのインスタンス、ダイナミック・オブジェクトの UTF-8 のシリアル化されたエンコード形式のいずれかです。 JSON オブジェクトの値を持つパラメータを要求に含める場合は、ダイナミック・オブジェクトを使用します。その 1 つのシナリオが OpenID Connect で定義されている `claims` パラメータです。ダイナミック・オブジェクトの詳細は、"JSON の使用" を参照してください。 `request` または `request_uri` パラメータを使用する場合は、"JWT としての要求オブジェクトの受け渡し" セクションを参照してください。 `code_verifier` パラメータを使用して、承認サーバに送信するシークレット PKCE 値を変更できます。既定では、PKCE シークレット は、SHA-256 ハッシュを使用したランダムな 43 文字の文字列から生成されます。カスタム文字列からシークレットを生成するには (例えば、128 文字を使用する場合)、`code_verifier` パラメータにカスタム値を設定します。

- このクライアントおよびセッションのアクセス・トークンがローカルに保存されている場合、出力として返される isAuthorized の値は 1 になります (スコープはチェックされていません)。それ以外の場合、このパラメータの値は 0 になります。さきほど IsAuthorized() メソッドを呼び出したので、この出力引数を確認する必要はありません。
- 出力として返される sc には、このメソッドが設定したステータス・コードが含まれます。
- responseMode は、承認サーバの応答モードを指定します。このモードは `"query"` (既定値)、`"fragment"`、または `"form_post"` にできます。ほとんどの場合、既定値が適切な値となります。
- sessionId はセッション ID を指定します。既定のセッション (%session.SessionId) をオーバーライドする場合のみこれを指定します。

注釈　Active Directory フェデレーション・サービス (ADFS) サーバは、承認エンドポイント URL にキーと値のペア `resource=urn:microsoft:userinfo` が含まれることを期待しています。`GetAuthorizationCodeEndpoint` の `properties` 引数を使用して、このキーと値のペアをサーバ記述で定義された URL の末尾に追加できます。管理ポータルで承認エンドポイントを変更してこの情報を含めることは避けてください。以下のコードを使用して、`GetAuthorizationCodeEndpoint` メソッドを呼び出す前に、`properties` 引数を変更してください。

```
set properties("resource") = "urn:microsoft:userinfo"
set url = ##class(%SYS.OAuth2.Authorization).GetAuthorizationCodeEndpoint(appName, scopes,
clientRedirectURI, .properties, .isAuthorized,.sc)
```

"バリエーション : OnPreHTTP 内でのリダイレクトの実行" も参照してください。

トークンの検証

アクセス・トークンと (必要に応じて) ID トークンを受け取ったクライアントは、追加のチェックを実行して、要求されたリソースの使用に必要な権限をユーザが保有しているかどうかを確認します。この検証を実行するために、クライアントは、ここで説明するメソッドを使用して追加情報を取得することができます。

ValidateIDToken()

場所:このメソッドはクラス %SYS.OAuth2.Validation にあります。

```
ClassMethod ValidateIDToken(applicationName As %String,
                            IDToken As %String,
                            accessToken As %String,
                            scope As %String,
                            aud As %String,
                            Output jsonObject As %RegisteredObject,
                            Output securityParameters As %String,
                            Output sc As %Status) As %Boolean
```

このメソッドは署名済みの OpenID Connect ID トークン (IDToken) を検証し、オブジェクト (jsonObject) を作成して ID トークンのプロパティを格納します。ID トークンを検証するために、このメソッドは、対象者 (aud が指定されている場合)、エンドポイント (サーバ記述で指定されているものとの一致が必要)、スコープ (scope が指定されている場合)、シグニチャを確認します。このメソッドは、ID トークンの有効期限が切れていないことも確認します。

さらに、ID トークンの `at_hash` プロパティに基づいてアクセス・トークン (accessToken) も検証します。

このメソッドは ID トークンが有効な場合は 1 を返し、それ以外の場合は 0 を返します。さらに、複数の引数を出力として返します。

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。
- IDToken は ID トークンです。
- accessToken はアクセス・トークンです。
- scope は、スペースで区切られたスコープのリストです。例えば、`"scope1 scope2 scope3"` のような形式をとります。
- aud は、このトークンを使用する対象者を指定します。トークンに関連付けられた `aud` プロパティがある場合 (通常、対象者はトークンの要求時に指定されているため)、`aud` はトークンの対象者と一致します。`aud` が指定されていない場合、対象者のチェックは行われません。
- 出力として返される jsonObject は、IDToken のプロパティを含むダイナミック・オブジェクトです。ID トークンは JWT です。ダイナミック・オブジェクトの詳細は、"JSON の使用" を参照してください。
- 出力として返される securityParameters は、ヘッダから取得されたセキュリティ情報を含む多次元配列です。このセキュリティ情報は、シグニチャや復号化の検証で必要に応じて追加使用されます。ValidateJWT() の securityParameters 引数を参照してください。
- 出力として返される sc には、このメソッドが設定したステータス・コードが含まれます。

このメソッドが成功 (1) を返す場合は、jsonObject を検証し、必要に応じて含まれているクレームを使用して、要求されたリソースへのアクセスを許可するかどうかを決定します。 必要に応じて securityParameters を使用します。

GetUserinfo()

場所:このメソッドはクラス %SYS.OAuth2.AccessToken にあります。

```
ClassMethod GetUserinfo(applicationName As %String,
                        accessToken As %String,
                        IDTokenObject As %RegisteredObject,
                        Output jsonObject As %RegisteredObject,
                        Output securityParameters As %String) As %Status
```

このメソッドはアクセス・トークンを Userinfo エンドポイントへ送信し、クレームを含む応答を受信し、このエンドポイントが返したクレームを含むオブジェクト (jsonObject) を作成します。 応答が JWT を返すとその応答は復号化され、署名が確認されてから jsonObject が作成されます。引数 IDTokenObject が指定されている場合、このメソッドは、Userinfo エンドポイントの sub クレームが IDTokenObject の sub クレームと一致していることも検証します。

要求は、指定されたアクセス・トークンによって承認されます。

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。
- accessToken はアクセス・トークンです。
- IDTokenObject (オプション) は、ID トークンを含むダイナミック・オブジェクトです。ダイナミック・オブジェクトの詳細は、"JSON の使用" を参照してください。
- 出力として返される jsonObject は、Userinfo エンドポイントが返したクレームを含むダイナミック・オブジェクトです。
- 出力として返される securityParameters は、ヘッダから取得されたセキュリティ情報を含む多次元配列です。このセキュリティ情報は、シグニチャや復号化の検証で必要に応じて追加使用されます。ValidateJWT() の securityParameters 引数を参照してください。

このメソッドが成功 (1) を返す場合は、jsonObject を検証し、必要に応じて含まれているクレームを使用して、要求されたリソースへのアクセスを許可するかどうかを決定します。 必要に応じて securityParameters を使用します。

HTTP 要求へのアクセス・トークンの追加

アクセス・トークンを受信して検証したクライアント・アプリケーションは、リソース・サーバへの HTTP 要求を作成できます。アプリケーションによっては、この HTTP 要求でアクセス・トークンを必要とすることもあります。

(ベアラー・トークン HTTP 承認ヘッダとして) アクセス・トークンを HTTP 要求へ追加するには、以下の手順を実行します。

1. %Net.HttpRequest のインスタンスを作成し、必要に応じてプロパティを設定します。

   このクラスの詳細は、"インターネット・ユーティリティの使用法" の" HTTP 要求の送信" を参照してください。

2. %SYS.OAuth2.AccessToken の AddAccessToken() メソッドを呼び出します。このメソッドにより、アクセス・トークンが HTTP 要求に追加されます。このメソッドは、以下のとおりです。

```
ClassMethod AddAccessToken(httpRequest As %Net.HttpRequest,
                           type As %String = "header",
                           sslConfiguration As %String,
                           applicationName As %String,
                           sessionId As %String) As %Status
```

   このメソッドは、指定されたアプリケーションおよびセッションに関連付けられているベアラー・アクセス・トークンを RFC 6750 で定義されているリソース・サーバへの要求に追加します。引数は以下のとおりです。

   - httpRequest は変更する %Net.HttpRequest のインスタンスです。
   - type は、HTTP 要求にアクセス・トークンを含める方法を指定します。
     - "header" ― ベアラー・トークン HTTP ヘッダを使用します。
     - "body" ― フォームエンコードされた本文を使用します。この場合の要求は、本文がフォームエンコードされた POST でなければなりません。
     - "query" ― クエリ・パラメータを使用します。

- sslConfiguration は、この HTTP 要求で使用する InterSystems IRIS SSL 構成です。これを省略すると、InterSystems IRIS はクライアント構成に関連付けられている SSL 構成を使用します。
- applicationName はクライアント・アプリケーションの名前です。
- sessionId はセッション ID を指定します。既定のセッション (%session.**SessionId**) をオーバーライドする場合のみこれを指定します。

このメソッドは、ステータス・コードを返すので、コードでこのステータス・コードを確認する必要があります。

3. ″インターネット・ユーティリティの使用法″ の ″HTTP 要求の送信″ の説明に従って HTTP 要求を送信します。そのためには、Get() や Put() などのメソッドを呼び出します。
4. 前の手順で返されたステータスを確認します。
5. HTTP 要求の **HttpResponse** プロパティとして利用できる HTTP 応答を必要に応じて検証します。

   ″インターネット・ユーティリティの使用法″ の ″HTTP 要求の送信″ を参照してください。

以下に例を示します。

ObjectScript

```
set httpRequest=##class(%Net.HttpRequest).%New()
// AddAccessToken adds the current access token to the request.
set sc=##class(%SYS.OAuth2.AccessToken).AddAccessToken(httpRequest,,"sslunittest",applicationName)
if $$$ISOK(sc) {
   set sc=httpRequest.Get("https://myresourceserver/csp/openid/openid.SampleResource.cls")
}
```

Web クライアントの代行認証の定義 (オプション)

必要に応じて、OAuth 2.0 クライアントとして使用する InterSystems IRIS Web クライアントの代行認証を定義できます。InterSystems IRIS には、これが可能な以下の 2 つの方法が用意されています。

- ZAUTHENTICATE ルーチンを作成して使用する方法。OAuth 2.0 で使用するために用意されているサンプルから作成します。クライアント・コードで %session.Login() を呼び出す必要もあります。
- カスタム・ログイン・ページを作成して使用する方法。ZAUTHENTICATE ルーチンを作成して使用する必要もあります (OAuth 2.0 で使用するために用意されている同じサンプルから作成)。ただし、クライアント・コードで %session.Login() を呼び出す必要はありません。

以降の項で詳しく説明します。最後のサブセクションでは、ZAUTHENTICATE のサンプルについて説明します。

″REST サービスの作成″ の ″REST サービスの保護″ にある ″REST アプリケーションおよび OAuth 2.0″ も参照してください。

重要　HealthShare® で認証を使用している場合は、インターシステムズが提供する ZAUTHENTICATE ルーチンを使用する必要があります。独自のルーチンは作成できません。

OAuth 2.0 クライアント用の ZAUTHENTICATE ルーチンの作成および使用

OAuth 2.0 クライアントとして使用する InterSystems IRIS Web クライアント用に ZAUTHENTICATE ルーチンを作成して使用するには、以下の手順をすべて実行します。

- クライアント・コードで、**%SYS.OAuth2.AccessToken** クラスの IsAuthorized() メソッドを呼び出してアクセス・トークンの取得に成功した後、%session 変数の Login() メソッドを呼び出します。ユーザ名として OAuth 2.0 のアプリケーション名を指定し、パスワードとして Web セッション ID を指定します。
- ZAUTHENTICATE ルーチンを作成します。このルーチンは、ロールや他のユーザ・プロパティの指定など、ユーザ・アカウントの基本的な設定を実行する必要があります。

インターシステムズが提供するサンプル・ルーチン OAUTH2.ZAUTHENTICATE.mac をコピーして変更することができます。このルーチンは GitHub の Samples-Security サンプルに含まれています (https://github.com/intersystems/Samples-Security)。"InterSystems IRIS で使用するサンプルのダウンロード" で説明されているようにサンプル全体をダウンロードすることもできますが、単に GitHub でルーチンを開いて、その内容をコピーするほうが簡単です。

ZAUTHENTICATE ルーチンを定義する場合、このルーチンは %SYS ネームスペース内にあり、ZAUTHENTICATE という名前である必要があります。"OAUTH2.ZAUTHENTICATE.mac サンプルの注意事項" を参照してください。代行認証に関する一般的な情報は、"代行 (ユーザ定義) 認証コードの作成" と "代行認証" を参照してください。

- [認証オプション] ページで、InterSystems IRIS インスタンスに対して代行認証を有効にします。

  この手順と次の手順の詳細は、"代行認証" を参照してください。

- 関連する Web アプリケーションに対して代行認証を有効にします。

OAuth 2.0 クライアント用のカスタム・ログイン・ページの作成および使用

OAuth 2.0 クライアントとして使用する InterSystems IRIS Web クライアント用にカスタム・ログイン・ページを作成して使用するには、以下の手順をすべて実行します。

- %OAuth2.Login のサブクラスを作成します。サブクラスで以下を実行します。
  - アプリケーション名、スコープ・リスト、およびレスポンス・モード (オプション) を指定します。以下のいずれかまたは両方を実行してこれらの項目を指定できます。
    - %OAuth2.Login のサブクラスのパラメータを指定。
    - DefineParameters() クラス・メソッドのオーバーライド。パラメータの指定と対照的に、このテクニックでは実行中にこれらの値を設定できます。

    これらのパラメータは、以下のとおりです。
    - APPLICATION ― これは、ログインするアプリケーションのアプリケーション名である必要があります。
    - SCOPE ― これは、アクセス・トークン要求に使用するスコープ・リストを指定します。これは、空白で区切られた文字列リストである必要があります。
    - RESPONSEMODE ― これは、レスポンスのモードを指定します。可能な値は "query" (既定値)、"fragment"、または "form_post" です。

    DefineParameters() クラス・メソッドには以下のシグニチャがあります。

    ```
    ClassMethod DefineParameters(Output application As %String, Output scope As %String, Output
    responseMode As %String)
    ```

    このメソッドは、アプリケーション名、スコープ・リスト、およびレスポンス・モードを出力引数として返します。このメソッドの既定の実装では、APPLICATION、SCOPE、および RESPONSEMODE の各クラス・パラメータの値が返されます。
  - %OAuth2.Login のサブクラスで、GetAccessTokenAuthorizationCode() 呼び出しのプロパティ・リストも指定します。そのためには、DefineProperties() クラス・メソッドをオーバーライドします。このメソッドには、以下のシグニチャがあります。

    ```
    ClassMethod DefineProperties(Output properties As %String)
    ```

    このメソッドは、properties 配列を出力として返します。これは、トークン要求に含める追加プロパティを指定するローカル配列です。properties 配列は以下の形式をとります。

ノード	値
properties(name)、ここで name はパラメータの名前です。	指定されたパラメータの値。

JSON オブジェクトである要求パラメータを追加するには、%DynamicObject のインスタンスであるプロパティ要素を作成します。または、UTF-8 でエンコードされたシリアル化オブジェクトである文字列を作成します。

`request` または `request_uri` 要求パラメータを追加するには、%SYS.OAuth2.Request クラスを使用して JWT を作成します。次に必要に応じて、properties`("request")` を JWT と等しい値に設定するかまたは properties`("request_uri")` を JWT の URL と同じ値に設定します。

- 関連 Web アプリケーションを構成して、カスタム・ログイン・ページを使用します。
- 前のセクションの説明に従って、ZAUTHENTICATE ルーチンを作成して使用します。ただし、中黒で示された最初の項目を除きます。(このシナリオでは、クライアント・コードで %session.Login() を呼び出す必要はありません。)

OAUTH2.ZAUTHENTICATE.mac サンプルの注意事項

OAUTH2.ZAUTHENTICATE.mac サンプル (https://github.com/intersystems/Samples-Security から取得) は、前のサブセクションで説明した両方のシナリオをサポートします。このサンプルでは、GetCredentials() サブルーチンは以下のようになります。

```
GetCredentials(ServiceName,Namespace,Username,Password,Credentials) Public {
    If ServiceName="%Service_WebGateway" {
        // Supply user name and password for authentication via a subclass of %OAuth2.Login
        Set Username="OAuth2"
        Set Password=$c(1,2,3)
    }
    Quit $$$OK
}
```

このサブルーチンは、ユーザ名とパスワードが指定されていないと呼び出されます (これはカスタム・ログイン・ページが使用された場合です)。サービス **%Service_WebGateway** では、このサンプルは、ユーザ名とパスワードを (後の処理で呼び出す) ZAUTHENTICATE() サブルーチンでも使用する特定の値に設定します。

ZAUTHENTICATE() サブルーチンには以下が含まれています。

```
If Username="OAuth2",Password=$c(1,2,3) {
    // Authentication is via a subclass of %OAuth2.Login that sets the query parameter CSPOAUTH2
    // with a hash value that allows GetCurrentApplication to determine the application --
    // username/password is supplied by GetCredentials.
    Set sc=##class(OAuth2.Response).GetCurrentApplication(.applicationName)
    Set sessionId=%session.SessionId
} Else {
    // If authentication is based on %session.Login, then application and session id are passed in.
    Set applicationName=Username
    Set sessionId=Password
}
```

後の手順では、次のように isAuthorized() メソッドを呼び出します。

```
Set
isAuthorized=##class(%SYS.OAuth2.AccessToken).IsAuthorized(applicationName,sessionId,,.accessToken,,,.error)
```

isAuthorized() が 1 を返す場合、後述のコードは、イントロスペクション・エンドポイントを呼び出し、そこから取得した情報を使用してユーザを定義します。

```
Set sc=##class(%SYS.OAuth2.AccessToken).GetIntrospection(applicationName,accessToken,.jsonObject)
...
Set Username="OAuth2"_jsonObject.sub
Set Properties("FullName")="OAuth account "_Username
Set Properties("Username")=Username
Set Properties("Password")=""    // we don't really care about oauth2 account password
// Set the roles and other Properties as appropriate.
Set Properties("Roles")=roles
```

コードで異なるロジックを使用して、ユーザの定義に必要な情報を取得できます。代わりに以下の方法でこの情報を取得することもできます。

- OpenID Connect を使用している場合には IDToken から取得。この場合、**%SYS.OAuth2.Validate** の ValidateIDToken() を呼び出します。
- OpenID Connect の場合には Userinfo エンドポイントから取得。この場合、**%SYS.OAuth2.AccessToken** の GetUserinfo() を呼び出します。

どの場合でも、ユーザ名が通常の InterSystems IRIS ユーザ名と一致しないユーザを定義する必要があります。

また、アプリケーションの要件に応じて、ルーチンでロールおよび Properties 配列のその他の部分を設定する必要もあります。"代行 (ユーザ定義) 認証コードの作成" と "代行認証" を参照してください。

アクセス・トークンの取り消し

承認サーバがトークンの取り消しをサポートする場合、管理ポータル経由またはプログラムによってアクセス・トークンを取り消すことができます。

ユーザのアクセス・トークンの取り消し

指定されたユーザのアクセス・トークンをすべて取り消すには、次の手順を実行します。

1. **[システム管理]→[セキュリティ]→[OAuth 2.0]→[管理]** を選択します。
2. ユーザ ID を **[ユーザのトークンの取り消し]** フィールドに入力します。
3. **[取り消す]** を選択します。

このタスクを実行するには、**%Admin_OAuth2_Registration** リソースの USE 権限を持つユーザとしてログインする必要があります。

プログラムによるアクセス・トークンの取り消し

クライアントがアクセス・トークンを取り消す必要がある場合は、**%SYS.OAuth2.AccessToken** の RevokeToken() メソッドを使用します。指定されたトークンを保持しているセッションが削除されると、システムはこのメソッドを自動的に呼び出します (取り消しエンドポイントが指定されている場合)。

```
ClassMethod RevokeToken(applicationName As %String, accessToken As %String) As %Status
```

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。
- accessToken はアクセス・トークンです。

要求は、`client_id` と `client_secret` が applicationName に関連付けられている、基本的な承認 HTTP ヘッダによって承認されます。

以下に例を示します。

```
set sc=##class(%SYS.OAuth2.AccessToken).RevokeToken("myclient",accessToken)
if $$$ISERR(sc) {
    //error handling here
}
```

サーバで取り消しエンドポイントが指定されていない場合や、**[クライアントの秘密鍵]** が指定されていない場合は、このメソッドを使用できません。

%SYS.OAuth2.AccessToken はメソッド RemoveAccessToken() も提供します。このメソッドはクライアントからアクセス・トークンを削除しますが、サーバからはアクセス・トークンを削除しません。

JWT で使用するキーの回転

ほとんどの場合、新しい公開/秘密鍵ペアをクライアントに生成させることができます。これは、非対称の RS256、RS384、および RS512 の各アルゴリズムに使用する RSA 鍵にのみ適用されます。(例外は、**[X509 証明書]** として、**[ダイナミック登録以外のソース]** を指定する場合です。この場合、新しいキーは生成できません。)

新しい公開/秘密鍵ペアの生成は、キーの回転と呼ばれています。このプロセスは、新しい秘密 RSA 鍵および関連する公開 RSA 鍵を秘密と公開の JWKS に追加します。

クライアントでキーの回転を実行すると、クライアントは新しい秘密 RSA 鍵を使用して、承認サーバに送信する JWT に署名します。同様に、クライアントは新しい公開 RSA 鍵を使用して、承認サーバに送信する JWT を暗号化します。クライアントは、承認サーバから受信した JWT を解読するために新しい RSA 鍵を使用しますが、これが失敗したら古い RSA 鍵を使用するため、古い公開 RSA 鍵を使用して作成された JWT を解読できます。最後に、承認サーバから受信した署名済みの JWT をクライアントが検証できない場合、クライアントに承認サーバの公開の JWKS の URL があると、クライアントは新しい公開の JWKS を取得し、署名の検証を再試行します。(クライアントが動的に登録された場合や、構成で **[URL から JWKS]** オプションが指定された場合、クライアントは承認サーバの公開の JWKS の URL を持っています。それ以外の場合にはクライアントはこの URL を持っていません。)

指定されたクライアント構成に対してキーを回転するには、次の手順を実行します。

1. 管理ポータルで **[システム管理]**→**[セキュリティ]**→**[OAuth 2.0]**→**[クライアント構成]** を選択します。
2. クライアント構成が関連付けられたサーバ記述を選択します。

   これによってシステムは、サーバ記述に関連付けられたクライアント構成をすべて表示します。
3. キーを回転させたいクライアントの構成を選択します。
4. **[キーの回転]** ボタンを選択します。

注釈　対称の HS256、HS384、および HS512 の各アルゴリズムは、常に対称鍵としてクライアントの秘密鍵を使用します。

クライアント用のキーの回転の API

クライアントでキーをプログラムによって回転させるには、**OAuth2.Client** の RotateKeys() メソッドを呼び出します。

新しい承認サーバの公開の JWKS を取得するには、**OAuth2.ServerDefinition** の UpdateJWKS() メソッドを呼び出します。

これらのメソッドの詳細は、クラス・リファレンスを参照してください。

承認サーバからの新しい公開の JWKS の取得

ほとんどの場合、承認サーバは JWKS の公開/秘密鍵ペアを生成します。公開の JWKS をクライアントが受信する方法はさまざまです。1 つの方法は、承認サーバが URL で公開の JWKS を提供する方法です。"サーバ記述の手動による作成 (Discovery なし)" の **[URL から JWKS]** オプションを参照してください。

承認サーバが **[URL から JWKS]** で定義されていて、承認サーバが JWKS の新しいペアを生成する場合、同じ URL から新しい公開の JWKS をクライアントに取得させることができます。そのためには、以下の操作を実行します。

1. 管理ポータルで **[システム管理]**→**[セキュリティ]**→**[OAuth 2.0]**→**[クライアント構成]** を選択します。
2. クライアント構成が関連付けられたサーバ記述を選択します。

   これによってシステムは、サーバ記述に関連付けられたクライアント構成をすべて表示します。
3. クライアントの構成を選択します。
4. **[JWKS の更新]** ボタンを選択します。

承認サーバが [URL から JWKS] で定義されておらず、承認サーバが JWKS の新しいペアを生成する場合、公開の JWKS を取得し、これをクライアントに送信し、ファイルからロードする必要があります。

OAuth 2.0 クライアントのバリエーション

この OAuth 2.0 クライアントとしての InterSystems IRIS® Web アプリケーションの使用法では、承認コード付与タイプに焦点を当てて説明します。

基本シナリオでは、クライアントは承認サーバからアクセス・トークンを受け取り、必要に応じて承認サーバの追加エンドポイント (イントロスペクション・エンドポイントか Userinfo エンドポイントまたはその両方) を呼び出します。その後、クライアントはリソース・サーバを呼び出します。リソース・サーバがこれらのエンドポイントを個別に呼び出すこともできます。

ここでは、いくつかのバリエーションについて説明します。

PKCE の無効化

既定では、承認コード付与タイプを使用するクライアントは、Proof Key for Code Exchange (PKCE) 拡張を活用します。ほとんどの場合、クライアントはこの既定の動作を変更するべきではありません。PKCE は広く受け入れられている重要なセキュリティ機能であるためです。ただし、まれに、PKCE を無効にしたいことがあります。

PKCE を無効にするには、メソッドを呼び出す前に、`GetAuthorizationCodeEndpoint()` の `properties` 引数を変更します。コードには、以下の行を含める必要があります。

```
Set properties("code_verifier")=""
```

`GetAuthorizationCodeEndpoint()` の `properties` 引数の詳細は、"メソッドの詳細" を参照してください。

暗黙付与タイプ

このバリエーションでは、クライアントはトークンの要求時に暗黙付与タイプを使用します。

構成要件 : "クライアントの構成" の手順を参照してください。ただし、[**クライアント・タイプ**] は使用事例に応じて指定します。

コード要件 : 全体の流れは承認コード付与タイプと似ていますが、GetAuthorizationCodeEndpoint() は呼び出しません。代わりに、**%SYS.OAuth2.Authorization** クラスの GetImplicitEndpoint() メソッドを呼び出します。

```
ClassMethod GetImplicitEndpoint(applicationName As %String,
                                scope As %String,
                                redirectURL As %String,
                                idtokenOnly As %Boolean = 0,
                                responseMode As %String,
                                ByRef properties As %String,
                                Output isAuthorized As %Boolean,
                                Output sc As %Status
                                sessionId as %String="") As %String
```

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。
- scope は、スペースで区切られた、アクセスが要求されているスコープのリストです。例えば、`"openid profile scope3 scope4"` のような形式をとります。

  既定値は、指定された applicationName のクライアント構成によって決まります。
- redirectURL は、承認サーバがクライアント・サーバにアクセス・トークンを返した後にブラウザをリダイレクトするページの URL です。
- idtokenOnly は ID トークンのみを取得可能にします。この引数が 0 の場合、メソッドはアクセス・トークンと ID トークン (要求に適切なスコープが含まれている場合) の両方を取得します。この引数が 1 の場合、メソッドはアクセス・トークンを取得しません。

- responseMode は、承認サーバの応答モードを指定します。このモードは "query" (既定値)、"fragment"、または "form_post" にできます。
- 参照によって渡される properties は、要求に追加されるパラメータを含む多次元配列です。この配列は以下の構造を持つ必要があります。

配列ノード	配列値
properties (parametername)、ここで parametername はパラメータの名前です。	指定されたパラメータの値。この値は、スカラ値、ダイナミック・オブジェクトのインスタンス、ダイナミック・オブジェクトの UTF-8 のシリアル化されたエンコード形式のいずれかです。 JSON オブジェクトの値を持つパラメータを要求に含める場合は、ダイナミック・オブジェクトを使用します。その 1 つのシナリオが OpenID Connect で定義されている claims パラメータです。ダイナミック・オブジェクトの詳細は、"JSON の使用" を参照してください。 request または request_uri パラメータを使用する場合は、"JWT としての要求オブジェクトの受け渡し" セクションを参照してください。

- このクライアントおよびセッションのアクセス・トークンがローカルに保存され、さらにそのアクセス・トークンが scope 引数で提供されるすべてのスコープを承認する場合、出力として返される isAuthorized の値は 1 になります。それ以外の場合、このパラメータの値は 0 になります。
- 出力として返される sc には、このメソッドが設定したステータス・コードが含まれます。
- sessionId はセッション ID を指定します。既定のセッション (%session.SessionId) をオーバーライドする場合のみこれを指定します。

"バリエーション : OnPreHTTP 内でのリダイレクトの実行" も参照してください。

パスワード資格情報付与タイプ

このバリエーションでは、クライアントはトークンの要求時にパスワード資格情報付与タイプを使用します。リソース所有者に属するパスワードがクライアントにある場合は、この付与タイプを使用できます。クライアント・アプリケーションは、ページのリダイレクトは行わずにトークン・エンドポイントへの HTTP POST 操作を実行できます。InterSystems IRIS は、これを行うメソッドを提供します。

構成要件 : "クライアントの構成" の手順を参照してください。ただし、[クライアントの秘密鍵] を指定する必要はありません。(一般に、クライアント秘密鍵は、クライアント秘密鍵を必要とし、かつこれを保護できる場合にのみ使用してください。)

コード要件 : アプリケーションは以下を実行します。

1. "トークンの取得" の説明に従って、%SYS.OAuth2.AccessToken の IsAuthorized() メソッドを呼び出し、戻り値 (およびエラーの有無) を確認します。
2. IsAuthorized() が 0 を返した場合は、%SYS.OAuth2.Authorization の GetAccessTokenPassword() メソッドを呼び出します。

```
ClassMethod GetAccessTokenPassword(applicationName As %String,
                                   username As %String,
                                   password As %String,
                                   scope As %String,
                                   ByRef properties As %String,
                                   Output error As %OAuth2.Error) As %Status
```

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。

- username はユーザ名です。
- password は対応するパスワードです。
- scope は、スペースで区切られた、アクセスが要求されているスコープのリストです。例えば、"scope1 scope2 scope3" のような形式をとります。

  既定値は、指定された applicationName のクライアント構成によって決まります。
- 参照によって渡される properties は、要求に追加されるパラメータを含む多次元配列です。この配列は以下の構造を持つ必要があります。

配列ノード	配列値
properties (parametername)、ここで parametername はパラメータの名前です。	指定されたパラメータの値。この値は、スカラ値、ダイナミック・オブジェクトのインスタンス、ダイナミック・オブジェクトの UTF-8 のシリアル化されたエンコード形式のいずれかです。 JSON オブジェクトの値を持つパラメータを要求に含める場合は、ダイナミック・オブジェクトを使用します。その 1 つのシナリオが OpenID Connect で定義されている claims パラメータです。ダイナミック・オブジェクトの詳細は、"JSON の使用" を参照してください。

- 出力として返される error は、Null であるか、エラー情報を含む %OAuth2.Error のインスタンスです。

このメソッドはトークン・エンドポイントへの HTTP POST 操作を実行し、アクセス・トークンを受信して保存します (存在する場合)。

3. error 引数を確認し、それに応じて処理を進めます。
4. "トークンの検証" と "HTTP 要求へのアクセス・トークンの追加" の説明に従って作業を進めます。

クライアント資格情報付与タイプ

このバリエーションでは、クライアントはトークンの要求時にクライアント資格情報付与タイプを使用します。この付与タイプを使用すると、クライアント・アプリケーションはユーザから独立してリソース・サーバと通信できます。ユーザ・コンテキストは存在しません。クライアント・アプリケーションは、ページのリダイレクトは行わずにトークン・エンドポイントへの HTTP POST 操作を実行できます。InterSystems IRIS は、これを行うメソッドを提供します。

構成要件："クライアントの構成" の手順を参照してください。[**クライアント・タイプ**] に [**プライベート**] を指定し、[**クライアント秘密鍵**] を指定します。

コード要件 ： アプリケーションは以下を実行します。

1. "トークンの取得" の説明に従って、%SYS.OAuth2.AccessToken の IsAuthorized() メソッドを呼び出し、戻り値 (およびエラーの有無) を確認します。
2. IsAuthorized() が 0 を返した場合は、%SYS.OAuth2.Authorization の GetAccessTokenClient() メソッドを呼び出します。

```
ClassMethod GetAccessTokenClient(applicationName As %String,
                                 scope As %String,
                                 ByRef properties As %String,
                                 Output error As %OAuth2.Error) As %Status
```

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。

- scope は、スペースで区切られた、アクセスが要求されているスコープのリストです。例えば、"scope1 scope2 scope3" のような形式をとります。

  既定値は、指定された applicationName のクライアント構成によって決まります。

- 参照によって渡される properties は、要求に追加されるパラメータを含む多次元配列です。前のサブセクションで説明した GetAccessTokenPassword() の properties 引数を参照してください。

- 出力として返される error は、Null であるか、エラー情報を含む %OAuth2.Error のインスタンスです。

このメソッドはトークン・エンドポイントへの HTTP POST 操作を実行し、アクセス・トークンを受信して保存します (存在する場合)。

3. error 引数を確認し、それに応じて処理を進めます。
4. "トークンの検証" と "HTTP 要求へのアクセス・トークンの追加" の説明に従って作業を進めます。

OnPreHTTP 内でのリダイレクトの実行

承認コードおよび暗黙付与タイプの場合、基本手順では以下に従います。

1. %SYS.OAuth2.AccessToken の IsAuthorized() メソッドを呼び出します。
2. GetAuthorizationCodeEndpoint() メソッド (承認コード付与タイプ用)、または GetImplicitEndpoint() メソッド (暗黙付与タイプ用) を呼び出します。
3. 上記の手順で返された URL を開くオプション (ボタンなど) を提供することで、ユーザは要求を承認することができます。

その代わりに、(アプリケーション内で) ページ・クラスの OnPreHttp() メソッドを変更することもできます。この場合、GetAccessTokenAuthorizationCode() メソッド (承認コード付与タイプ用) または GetAccessTokenImplicit() メソッド (暗黙付与タイプ用) を呼び出します。このメソッドは、最初にページのコンテンツを表示せずに、承認サーバの認証フォームにブラウザを直接移動します (必要な場合)。

JWT としての要求オブジェクトの受け渡し

OpenID Connect Core 仕様のセクション 6 で規定されているように、InterSystems IRIS は JWT としての要求オブジェクトの受け渡しもサポートしています。値または参照によって要求オブジェクトを渡すことができます。

いずれの場合も、%SYS.OAuth2.Request クラスのメソッドを使用します。このセクションに記載されていないその他のメソッドについては、クラス・リファレンスを参照してください。

値による要求オブジェクトの受け渡し

request パラメータを使用して JWT として要求オブジェクトを渡すには、以下の手順を実行します。

1. %SYS.OAuth2.Request クラスの MakeRequestJWT() メソッドを呼び出します。

```
ClassMethod MakeRequestJWT(applicationName As %String,
                           ByRef properties As %String,
                           Output sc As %Status) As %String
```

以下はその説明です。

- applicationName はクライアント・アプリケーションの名前です。
- 参照によって渡される properties は、要求に追加されるパラメータを含む多次元配列です。この配列は以下の構造を持つ必要があります。

配列ノード	配列値
properties (parametername)、ここで parametername はパラメータの名前です。	指定されたパラメータの値。この値は、スカラ値、ダイナミック・オブジェクトのインスタンス、ダイナミック・オブジェクトの UTF-8 のシリアル化されたエンコード形式のいずれかです。

- 出力として返される sc には、このメソッドが設定したステータス・コードが含まれます。

このメソッドは JWT の文字列を返します。以下に例を示します。

ObjectScript

```
// create jwt
set jwt=##class(%SYS.OAuth2.Request).MakeRequestJWT("myapp",.properties,.sc)
```

2. GetAuthorizationCodeEndpoint() または GetImplicitEndpoint() の引数として使用する properties 配列を変更します。前の手順で作成した JWT と等しくなるようにノード properties`("request")` を設定します。以下に例を示します。

ObjectScript

```
set properties("request")=jwt
```

3. GetAuthorizationCodeEndpoint() または GetImplicitEndpoint() を呼び出すときに properties 配列を指定します。以下に例を示します。

ObjectScript

```
set url=##class(%SYS.OAuth2.Authorization).GetAuthorizationCodeEndpoint("myapp",
      scope,redirect,.properties,.isAuthorized,.sc, responseMode)
```

参照による要求オブジェクトの受け渡し

`request_uri` パラメータを使用して JWT として要求オブジェクトを渡すには、以下の手順を実行します。

1. %SYS.OAuth2.Request クラスの UpdateRequestObject() メソッドを呼び出します。

```
ClassMethod UpdateRequestObject(applicationName As %String,
                                requestName As %String,
                                ByRef properties As %String,
                                Output sc As %Status) As %SYS.OAuth2.Request
```

以下はその説明です。

- applicationName はクライアント・アプリケーションの名前です。
- requestName は要求の名前です。
- 参照によって渡される properties は、要求に追加されるパラメータを含む多次元配列です。この配列は以下の構造を持つ必要があります。

配列ノード	配列値
properties (parametername)、ここで parametername はパラメータの名前です。	指定されたパラメータの値。この値は、スカラ値、ダイナミック・オブジェクトのインスタンス、ダイナミック・オブジェクトの UTF-8 のシリアル化されたエンコード形式のいずれかです。

- 出力として返される sc には、このメソッドが設定したステータス・コードが含まれます。

このメソッドは %SYS.OAuth2.Request のインスタンスの作成、保存、返却を行います。

ObjectScript

```
 // create requestobject
 set
requestobject=##class(%SYS.OAuth2.Request).UpdateRequestObject("myapp","myrequest",.properties,.sc)
```

2. 保存されている要求オブジェクトの URL を取得します。そのためには、このインスタンスの GetURL() メソッドを呼び出します。GetURL() は、最初の引数の出力としてステータス・コードを返すので、コードでこれを確認します。

   ObjectScript

   ```
    Set requesturl=requestobject.GetURL()
   ```

3. GetAuthorizationCodeEndpoint() または GetImplicitEndpoint() の引数として使用する properties 配列を変更します。前の手順で取得した URL と等しくなるようにノード properties("request_uri") を設定します。以下に例を示します。

   ```
   set properties("request_uri")=requesturl
   ```

4. GetAuthorizationCodeEndpoint() または GetImplicitEndpoint() を呼び出すときに properties 配列を指定します。以下に例を示します。

   ```
    set url=##class(%SYS.OAuth2.Authorization).GetAuthorizationCodeEndpoint("myapp",
          scope,redirect,.properties,.isAuthorized,.sc, responseMode)
   ```

承認サーバの他のエンドポイントの呼び出し

%SYS.OAuth2.Authorization のメソッドを使用すると、承認サーバの特定のエンドポイント・セットを呼び出すことができます。承認サーバに他のエンドポイントがある場合は、以下の一般的なプロセスを使用してそのエンドポイントを呼び出します。

1. %Net.HttpRequest のインスタンスを作成し、そのプロパティの設定とメソッドの呼び出しを必要に応じて行い、要求を定義します。

   ```
   Set httpRequest=##class(%Net.HttpRequest).%New()
   Set httpRequest.ContentType="application/x-www-form-urlencoded"
   ...
   ```

   このクラスの詳細は、"インターネット・ユーティリティの使用法" の" HTTP 要求の送信" を参照してください。

2. この要求に認証を追加するには、%SYS.OAuth2.AccessToken の AddAuthentication() メソッドを呼び出します。

   ```
   ClassMethod AddAuthentication(applicationName As %String, httpRequest As %Net.HttpRequest) As
   %Status
   ```

   以下はその説明です。

   - applicationName は OAuth 2.0 クライアントの名前です。
   - httpRequest は %Net.HttpRequest のインスタンスです。

   InterSystems IRIS は指定されたクライアントを検索し、その [認証タイプ]、[SSL 構成]、その他の情報を使用して適切な認証を要求に追加します。

3. 必要に応じてクライアント構成を開くことで、そこに含まれるプロパティを使用できます。そのためには、%SYS ネームスペースに切り替えて、OAuth2.Client の Open() メソッドを呼び出し、クライアント名を引数として渡します。

ObjectScript

```
New $NAMESPACE
set $NAMESPACE="%SYS"
Set client=##class(OAuth2.Client).Open(applicationName,.sc)
If client="" Quit
```

4. HTTP 要求オブジェクトの Post()、Get()、または Put() メソッドを必要に応じて呼び出し、承認サーバのトークン・エンドポイントを引数として提供します。以下に例を示します。

ObjectScript

```
set sc=httpRequest.Post(client.ServerDefinition.TokenEndpoint)
```

5. 必要に応じて追加処理を実行します。

OAuth 2.0 リソース・サーバとしての InterSystems IRIS Web アプリケーションの使用法

このページでは、OAuth 2.0 フレームワークを使用するリソース・サーバとして InterSystems IRIS® の Web アプリケーションを使用する方法について説明します。

このページでは、リソース・サーバが承認サーバのイントロスペクション・エンドポイントを使用するシナリオについて主に説明します。バリエーションの詳細は、前のセクションを参照してください。

JWT の署名、暗号化、シグニチャの検証、および解読に使用するキーの回転のプロセスについては別途説明します。

InterSystems IRIS リソース・サーバの前提条件

このページに記載されているタスクを開始する前に、以下のものが利用可能であることを確認します。

- OAuth 2.0 承認サーバ。
- リソース・サーバが承認サーバのエンドポイントを使用する場合は、そのリソース・サーバを OAuth 2.0 承認サーバのクライアントとして登録することができます。詳細は、承認サーバの実装環境によって異なります。

  この場合は、後でこのサーバの具体的な詳細情報も必要になります。

  - 承認サーバの場所 (発行者エンドポイント)
  - トークン・エンドポイントの場所
  - Userinfo エンドポイントの場所 (サポートされている場合。OpenID Connect Core を参照)
  - トークン・イントロスペクション・エンドポイントの場所 (サポートされている場合。RFC 7662 を参照)
  - トークン取り消しエンドポイントの場所 (サポートされている場合。RFC 7009 を参照)
  - 承認サーバによるダイナミック登録のサポートの有無

- 承認サーバがダイナミック登録をサポートしない場合、リソース・サーバのクライアント ID およびクライアント秘密鍵が必要です。承認サーバがこの 2 つの情報を生成します (1 回限り)。ユーザは、この情報をリソース・サーバ・マシンへ安全に伝える必要があります。

構成要件

"構成要件" を参照してください。クライアント構成の作成手順を以下のように変更します。

- **[アプリケーション名]** にはリソース・サーバのアプリケーション名を指定します。
- **[クライアント・タイプ]** には **[リソース・サーバ]** を指定します。

  [リソース・サーバ] をタイプとして指定した場合、構成ページにはリソース・サーバに適用されるオプションのみが表示されることに注意してください。

- [ClientID] にはリソース・サーバのクライアント ID を使用します。
- [clientSecret] にはリソース・サーバのクライアント秘密鍵を使用します。

コード要件

OAuth 2.0 リソース・サーバは要求を受け取り、そこに含まれるアクセス・トークンを検証し、(そのアクセス・トークンに応じて) 要求された情報を返します。

InterSystems IRIS リソース・サーバを作成するには、リソース・サーバの Web アプリケーションが使用するネームスペースの %CSP.REST のサブクラスを作成します。このクラスでは、URL マップおよび対応するメソッドを作成します。これらメソッドでは、以下を実行します。

1. %SYS.OAuth2.AccessToken の GetAccessTokenFromRequest() メソッドを呼び出します。このメソッドは、以下のとおりです。

   ```
   ClassMethod GetAccessTokenFromRequest(Output sc As %Status) As %String
   ```

   このページが受信した HTTP 要求にアクセス・トークンがある場合、メソッドはそれを返します。RFC 6750 にある 3 つのフォーマットの 1 つが使用されます。出力として返されるパラメータ sc は、エラー検出の有無を示すステータス・コードです。要求で SSL/TLS を使用しなかった場合はエラーが発生します。さらに、要求に有効なベアラー・ヘッダが含まれていなかった場合にもエラーが発生します。

2. ステータス・コードがエラーであるかどうかを確認します。

   ステータスがエラーの場合、メソッドは適切なエラーを返します (要求された情報は返しません)。

3. ステータス・コードがエラーでない場合は、アクセス・トークンを検証します。そのためには、ValidateJWT() を使用するか、独自のカスタム・メソッドを使用します。"メソッドの詳細" を参照してください。

4. 必要に応じて、追加情報を得るために GetIntrospection() メソッドを呼び出します。このメソッドは承認サーバのイントロスペクション・エンドポイントを呼び出し、アクセス・トークンに関するクレームを取得します。このメソッドは、以下のとおりです。

   ```
   ClassMethod GetIntrospection(applicationName As %String,
                                accessToken As %String,
                                Output jsonObject As %RegisteredObject) As %Status
   ```

   引数は以下のとおりです。

   - applicationName はクライアント・アプリケーションの名前です。
   - accessToken は以前に返されたアクセス・トークンです。
   - 出力として返される jsonObject は、承認サーバがこのアクセス・トークンに関して作成したクレームを含む JSON オブジェクトです。

5. 情報に対するユーザの要求を認可すべきであることを前述の手順が示している場合は、要求された処理を実行し、要求された情報を返します。

以下に例を示します。

```
// This is a dummy resource server which just gets the access token from the request
// and uses the introspection endpoint to ensure that the access token is valid.
// Normally the response would not be security related, but would ocntain some interesting
// data based on the request parameters.
set accessToken=##class(%SYS.OAuth2.AccessToken).GetAccessTokenFromRequest(.sc)
if $$$ISOK(sc) {
   set sc=##class(%SYS.OAuth2.AccessToken).GetIntrospection("demo resource",accessToken,.jsonObject)

    if $$$ISOK(sc) {
        write "OAuth 2.0 access token used to authorize resource server (RFC 6749)<br>"
        write "Access token validated using introspection endpoint (RFC 7662)<br>"
        write "   scope='"_jsonObject.scope_"'<br>"
        write "   user='"_jsonObject.username_"'",!
```

```
        } else {
            write "Introspection Error="_..EscapeHTML($system.Status.GetErrorText(sc)),!
        }
    } else {
        write "Error Getting Access Token="_$system.Status.GetErrorText(sc),!
    }

    Quit $$$OK
```

トークンの検証

アクセス・トークンを受け取ったリソース・サーバは、追加のチェックを実行して、要求されたリソースの使用に必要な権限をユーザが保有しているかどうかを確認します。この検証を実行するために、クライアントは、ここで説明するメソッドを使用して追加情報を取得することができます。

ValidateJWT()

場所:このメソッドはクラス %SYS.OAuth2.Validation にあります。

```
ClassMethod ValidateJWT(applicationName As %String,
                        accessToken As %String,
                        scope As %String,
                        aud As %String,
                        Output jsonObject As %RegisteredObject,
                        Output securityParameters As %String,
                        Output sc As %Status) As %Boolean
```

アクセス・トークンが (opaque トークンではなく) JWT である場合にのみこのメソッドを使用します。

このメソッドは JWT の復号化 (必要な場合) と検証を行い、オブジェクト (jsonObject) を作成して JWT プロパティを格納します。JWT を検証するために、このメソッドは、対象者 (aud が指定されている場合)、発行者エンドポイント (サーバ記述で指定されているものとの一致が必要)、スコープ (scope が指定されている場合) を確認します。このメソッドは、アクセス・トークンの有効期限が切れていないことも確認します。署名された JWT と署名されていない JWT の両方が許可されます。JWT が署名されている場合は、そのシグニチャを確認します。

このメソッドは JWT が有効な場合は 1 を返し、それ以外の場合は 0 を返します。さらに、複数の引数を出力として返します。

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。
- accessToken は検証される JWT です。
- scope は、スペースで区切られたスコープのリストです。例えば、"scope1 scope2 scope3" のような形式をとります。

  scope が指定されている場合は、このスコープを含むスコープ・クレームが JWT に含まれている必要があります。
- aud は、このトークンを使用する対象者を指定します。トークンに関連付けられた aud プロパティがある場合 (通常、対象者はトークンの要求時に指定されているため)、aud はトークンの対象者と一致します。aud が指定されていない場合、対象者のチェックは行われません。
- 出力として返される jsonObject は、JWT のクレームを含むダイナミック・オブジェクトです。このダイナミック・オブジェクトには aud、exp、iss などのプロパティが含まれます。ダイナミック・オブジェクトの詳細は、"JSON の使用" を参照してください。
- 出力として返される securityParameters は、ヘッダから取得されたセキュリティ情報を含む多次元配列です。このセキュリティ情報は、シグニチャや復号化の検証で必要に応じて追加使用されます。

  この配列には以下のノードがあります。

ノード	値
securityParameters("sigalg")	シグニチャまたは MAC アルゴリズム。JWT が署名されている場合にのみ設定します。
securityParameters("keyalg")	キー管理アルゴリズム
securityParameters("encalg")	コンテンツ暗号化アルゴリズム

keyalg と encalg のノードは、両方が指定されるか、両方が Null になります。

- 出力として返される sc には、このメソッドが設定したステータス・コードが含まれます。

このメソッドが成功 (1) を返す場合は、jsonObject を検証し、必要に応じて含まれているクレームを使用して、要求されたリソースへのアクセスを許可するかどうかを決定します。必要に応じて securityParameters を使用します。

Oauth 仕様では、アプリケーションは署名された JWT と署名されていない JWT の両方を受け入れることができるため、ValidateJWT メソッドは署名されていない JWT を拒否しません。ただし、多くの場合、署名されていない JWT を拒否して、アプリケーションにより厳しいセキュリティを実装することを強くお勧めします。メソッドから返された securityParameters 配列を調べることで、ValidateJWT に渡されたトークンが署名されていないかどうかを確認できます。securityParameters("sigalg") が設定されていない場合、トークンは署名されていません。例えば、以下のコードは、トークンが署名されていないかどうかを判断し、署名されていない場合は拒否します。

```
Set tInitialValidationPassed = ##class(%SYS.OAuth2.Validation).ValidateJWT(tClientName,
tAccessToken, "", "", .tJsonObj,.tSecurityParams, .tValidateStatus)
// the “sigalg” subscript is set only if the JWT was signed
Set tIsTokenSigned = $Data(tSecurityParams("sigalg"))#2
If 'tIsTokenSigned {
   $$$ThrowStatus($System.Status.Error($$$AccessDenied))
}
```

GetIntrospection()

場所:このメソッドはクラス %SYS.OAuth2.AccessToken にあります。

```
ClassMethod GetIntrospection(applicationName As %String,
                             accessToken As %String,
                             Output jsonObject As %RegisteredObject) As %Status
```

このメソッドはアクセス・トークンをイントロスペクション・エンドポイントへ送信し、クレームを含む応答を受信し、このエンドポイントが返したクレームを含むオブジェクト (jsonObject) を作成します。

要求は、client_id と client_secret が applicationName に関連付けられている、基本的な承認 HTTP ヘッダによって承認されます。

引数は以下のとおりです。

- applicationName はクライアント・アプリケーションの名前です。
- accessToken はアクセス・トークンです。
- 出力として返される jsonObject は、イントロスペクション・エンドポイントが返したクレームを含むダイナミック・オブジェクトです。ダイナミック・オブジェクトの詳細は、"JSON の使用" を参照してください。

サーバでイントロスペクション・エンドポイントが指定されていない場合や、[クライアント秘密鍵] が指定されていない場合は、このメソッドを使用できません。

このメソッドが成功 (1) を返す場合は、jsonObject を検証し、必要に応じて含まれているクレームを使用して、要求されたリソースへのアクセスを許可するかどうかを決定します。

バリエーション

このページでは、InterSystems IRIS リソース・サーバが承認サーバのイントロスペクション・エンドポイントを使用するシナリオについて主に説明します。このセクションでは、使用可能ないくつかのバリエーションについて説明します。

バリエーション:リソース・サーバによる Userinfo エンドポイントの呼び出し

リソース・サーバは Userinfo エンドポイントを呼び出すこともできます。そのためには、OAuth クライアントの場合と同様に、リソース・サーバのコードで GetUserinfo() メソッドを使用する必要があります。

バリエーション:リソース・サーバでエンドポイントを呼び出さない

InterSystems IRIS リソース・サーバが承認サーバのエンドポイントを使用しない場合は、このマシンで OAuth 2.0 構成を作成する必要はありません。

さらに、リソース・サーバが GetAccessTokenFromRequest() を使用する必要はありません。代わりに、リソース・サーバは HTTP 承認ヘッダからアクセス・トークンを直接取得し、必要に応じて使用することができます。

OAuth 2.0 承認サーバとしての InterSystems IRIS の使用法

このページでは、OAuth 2.0 承認サーバとして InterSystems IRIS® インスタンスを使用する方法について説明します。

クライアント定義を作成するユーザとサーバを設定するユーザは異なる可能性が高いためです。また、クライアント定義の継続的な作成が必要になる場合があるためです。そのため、クライアント定義の作成タスクは、独立したセクションとしてこの記事の最後に収録しています。

InterSystems IRIS 承認サーバの構成要件

OAuth 2.0 承認サーバとして InterSystems IRIS インスタンスを使用するには、次の構成タスクを実行します。

- InterSystems IRIS にサービスを提供している Web サーバでは、その Web サーバが SSL を使用するように構成します。SSL を使用するように Web サーバを構成する方法は、このドキュメントの対象外であるため、ここでは取り上げません。
- サーバが使用する InterSystems IRIS の SSL 構成を作成します。

  これはクライアント SSL 構成です。証明書は不要です。この構成は Web サーバへの接続に使用されます。この接続を通じて、承認サーバは、`request_uri` パラメータが指定する要求オブジェクトにアクセスします。この接続を通じて、承認サーバは、クライアントの JWKS を更新するときに `jwks_uri` にもアクセスします。クライアントが `request_uri` パラメータを使用して要求を送信せず、承認サーバが `jwks_uri` パラメータでクライアントの JWKS を更新しない場合、承認サーバは SSL 構成を必要としません。

  SSL 構成の作成の詳細は、インターシステムズの "TLS ガイド" を参照してください。

  それぞれの SSL 構成には一意の名前が付いています。参照用に、このドキュメントではこの SSL 構成を `sslconfig` と呼びますが、一意の名前を任意に付けることができます。
- 以下のサブセクションの説明に従ってサーバ構成を作成します。
- その後、必要に応じてクライアント定義を作成します。前のセクションを参照してください。

承認サーバの構成

このタスクを実行するには、**`%Admin_OAuth2_Server`** リソースの USE 権限を持つユーザとしてログインする必要があります。

1. 管理ポータルで **[システム管理]**→**[セキュリティ]**→[OAuth 2.0]→**[サーバ構成]** を選択します。
2. **[一般]** タブで、以下の詳細を指定します。
   - **[説明]** ― 必要に応じて説明を入力します。

- **[発行者エンドポイント]** – 承認サーバのエンドポイント URL を指定します。この URL を指定するには、次のオプションの値を入力します。
  - **[ホスト名]** – 承認サーバのホスト名または IP アドレスを指定します。
  - **[ポート]** – Web ゲートウェイ構成の変更に対応するために必要な場合は、この値を指定します。
  - **[接頭語]** – Web ゲートウェイ構成の変更に対応するために必要な場合は、この値を指定します。

  指定した発行者エンドポイントは以下の形式になります。

  ```
  https://hostname:port/prefix/oauth2
  ```

  [ポート] の指定を省略すると、コロンが省略されます。同様に、**[接頭語]** の指定を省略すると、`hostname:port` と `oauth2` の間のスラッシュが 1 つだけになります。
- **[対象者は必須]** – 承認サーバが承認コードおよび暗黙の要求で `aud` パラメータを必要とするかどうかを指定します。このチェック・ボックスのチェックを外すと、`aud` は必須でなくなります。
- **[ユーザセッションをサポート]** – 承認サーバがユーザ・セッション (Web セッションではない) をサポートするかどうかを指定します。選択すると、InterSystems IRIS はセッション維持クラス (既定のセッション維持クラスは OAuth2.Server.Session ですが、カスタマイズできます。詳細は、"コードのカスタマイズ・オプション" を参照してください) を使用します。このチェック・ボックスのチェックを外すと、ユーザ・セッションはサポートされません。
- **[パブリッククライアント更新を許可]** – サーバでクライアントのリフレッシュ・トークンを処理できるようにするかどうかを指定します。このチェック・ボックスにチェックを付けると、サーバはリフレッシュ・トークンを処理するためにクライアントの秘密鍵を必要としなくなります。
- [Enforce Proof Key for Code Exchange (PKCE) for public clients] – 承認サーバで承認コード付与タイプがサポートされる場合、このチェック・ボックスにチェックを付けると、パブリック・クライアントは承認コードを要求してコードをアクセス・トークンに交換する際に、PCKE シークレット値を提供する必要があります。
- [Enforce Proof Key for Code Exchange (PKCE) for confidential clients] – 承認サーバで承認コード付与タイプがサポートされる場合、このチェック・ボックスにチェックを付けると、機密クライアントは承認コードを要求してコードをアクセス・トークンに交換する際に、PCKE シークレット値を提供する必要があります。
- **[HTTPベースのフロントチャネルログアウトをサポート]** – フロント・チャネル・ログアウトを有効にするか無効にするかを指定します。既定では、フロント・チャネル・ログアウトが有効になっています。このチェック・ボックスのチェックを外すと、ユーザがサーバからログアウトしたときに、そのユーザはサーバに登録されているクライアントからもログアウトされます。

  注釈　InterSystems IRIS 承認サーバでフロント・チャネル・ログアウトがサポートされるようにするには、`/oauth2` Web アプリケーションの **[ユーザ Cookie スコープ]** を `[Lax]` に設定する必要があります。アプリケーションの設定の構成の詳細は、"アプリケーションの作成および編集" を参照してください。
- **[フロントチャネルログアウトURLとともに sid (セッションID) クレームの送信をサポート]** – フロント・チャネル・ログアウト URL と共にセッション ID クレームを送信できるようにするかどうかを指定します。既定では、セッション ID が送信されるようになっています。このチェック・ボックスのチェックを外すと、サーバがフロント・チャネル・ログアウト URL を呼び出した場合、その URL には `iss` (発行者) と `sid` (セッション ID) の各クエリ・パラメータが付加されません。
- **[リフレッシュ・トークンの返却]** – アクセス・トークンと共にリフレッシュ・トークンが返される条件を指定します。それぞれのビジネス・ケースに適したオプションを選択します。
- **[サポートされている付与タイプ]** – この承認サーバがアクセス・トークンを作成するために使用を許可する付与タイプを指定します。少なくとも 1 つ選択します。
- **[OpenID プロバイダ・ドキュメント]** – OpenID プロバイダによって提供される URL を以下のように指定します。

- **[サービス・ドキュメント URL]** – OpenID プロバイダの使用時に開発者が理解しておく必要があると思われる、人が読める情報を提供する Web ページの URL です。
- **[ポリシー URL]** – Relying Party がプロバイダによって提供されるデータを使用する方法について、OpenID プロバイダのポリシーを記述する Web ページの URL です。
- **[サービス条件 URL]** – OpenID プロバイダのサービス条件を記述する Web ページの URL です。

- **[SSL/TLS 構成]** – 承認サーバ用に作成した SSL/TLS 構成を選択します (例:`sslconfig`)。

3. **[スコープ]** タブで、以下の詳細を指定します。
   - **[スコープ]** と **[説明]** の列を持つテーブル – この承認サーバがサポートしているスコープをすべて指定します。
   - **[サポートされていないスコープを許可]** – サポートしていないスコープ値を承認サーバが無視するかどうかを指定します。このチェック・ボックスのチェックを外すと、サポートされていないスコープ値が要求に含まれる場合、承認サーバはエラーを返します。この場合、要求は処理されません。このチェック・ボックスにチェックを付けると、承認サーバはスコープを無視して要求を処理します。
   - **[既定のスコープ]** – アクセス・トークンの要求やクライアント構成でスコープが指定されていない場合に使用する、アクセス・トークンの既定のスコープを指定します。

4. **[間隔]** タブで、以下の詳細を指定します。
   - **[アクセス・トークン間隔]** – このサーバが発行したアクセス・トークンの有効期限が切れる秒数を指定します。デフォルトは 3600 秒です。
   - **[認証コードの間隔]** – このサーバが発行した認証コードの有効期限が切れる秒数を指定します。デフォルトは 60 秒です。
   - **[トークン間隔を更新]** – このサーバが発行したリフレッシュ・トークンの有効期限が切れる秒数を指定します。既定値は 24 時間 (86,400 秒) です。
   - **[セッション終了間隔]** – ユーザ・セッションが自動的に終了する秒数を指定します。値 0 はセッションが自動的に終了しないことを意味します。既定値は 24 時間 (86,400 秒) です。
   - **[クライアント秘密鍵の有効期間]** – このサーバが発行したクライアント秘密鍵の有効期限が切れる秒数を指定します。既定値 (0) は、クライアント秘密鍵が期限切れにならないことを示します。

5. **[JWT 設定]** タブで、以下の詳細を指定します。
   - **[X509 資格情報から JWT 設定を作成]** – 証明書に関連付けた秘密鍵を署名と暗号化で使用する場合は、このオプションを選択します。この場合は、"証明書と JWT (JSON Web Token)" の "OAuth 2.0 承認サーバの証明書の使用" も参照してください。

     注釈　インターシステムズでは、**[X509 資格情報から JWT 設定を作成]** オプションが使用されることはまれで、次に説明する既定の動作が代わりに使用されると考えています。

     このオプションのチェックを外したままにすると、システムは JWKS (JSON Web Key Set) のペアを生成します。一方の JWKS は秘密鍵で、(アルゴリズムあたり) 必要なすべての秘密鍵だけでなく、対称鍵として使用するクライアント秘密鍵も含んでいます。この JWKS は共有されません。他方の JWKS には、対応する公開鍵が含まれていて、公開されて利用できます。また、InterSystems IRIS は、公開の JWKS をクライアントにコピーして、クライアントが承認サーバから JWT の暗号化およびシグニチャの検証を実行できるようにします。
   - **[署名アルゴリズム]** – JWT を署名するときに使用するアルゴリズムを選択します。JWT を署名しない場合は空白のままにします。
   - **[キー管理アルゴリズム]** – JWT を暗号化するときにキー管理で使用するアルゴリズムを選択します。

     この選択は、コンテンツの暗号化アルゴリズムを選択する場合にのみ行います。

- **[コンテンツ暗号化アルゴリズム]** – JWT を暗号化するときに使用するアルゴリズムを選択します。JWT を暗号化しない場合は空白のままにします。アルゴリズムを選択する場合は、キー管理のアルゴリズムも選択する必要があります。

6. **[カスタマイズ]** タブで、"コードのカスタマイズ・オプション" に従って詳細を指定します。
7. **[保存]** を選択します。

この構成を保存すると、システムは承認サーバが使用する Web アプリケーション (`/oauth2`) を作成します。この Web アプリケーションは変更しないでください。

コードのカスタマイズ・オプションと全体的な手順

このセクションでは、承認サーバの構成オプションの**[カスタマイズ・オプション]** セクションにある項目について説明します。サブセクションでは、全体的な手順と既定のクラスについて説明します。

- **[認証クラス]** – %OAuth2.Server.Authenticate (既定値) またはそのクラスのカスタム・サブクラスを使用します。

  カスタム・サブクラスを定義する場合は、必要に応じて以下のメソッドの一部またはすべてを実装します。

  - BeforeAuthenticate() – 必要に応じてこのメソッドを実装し、認証の前にカスタム処理を実行します。
  - DisplayLogin() – 必要に応じてこのメソッドを実装し、ユーザを識別するログイン・ページを表示します。(一般的でない代替メソッドについては、"DirectLogin() の実装" を参照してください。)
  - DisplayPermissions() – 必要に応じてこのメソッドを実装し、要求された許可をユーザに表示します。
  - AfterAuthenticate() – 必要に応じてこのメソッドを実装し、認証の後にカスタム処理を実行します。

- **[ユーザ検証クラス]** – %OAuth2.Server.Validate (既定値) を使用するか、次のメソッドを定義するカスタム・クラスを使用します。

  - ValidateUser() (クライアント資格情報を除くすべての付与タイプで使用)
  - ValidateClient() (クライアント資格情報付与タイプで使用)

  カスタム・クラスを定義して使用することを強くお勧めします。%OAuth2.Server.Validate クラスは実例で説明するためのものであり、実稼働環境での使用に適していない可能性があります。

- **[セッション維持クラス]** – 既定のセッション維持クラス OAuth2.Server.Session は、HTTP 専用 Cookie を介してユーザ・セッションを維持します。この既定のクラスを拡張してカスタム・ロジックを実装することはできません。代わりに、`%OAuth2.Server.CookieSession` または `%OAuth2.Server.AbstractSession` を要件に応じて拡張できます。既定のクラスのロジックの大部分は `%OAuth2.Server.CookieSession` が元になっているため、既存のコードを活用するカスタムのセッション維持クラスを実装する場合は (opaque ブラウザ Cookie の使用を含む)、%OAuth2.Server.CookieSession を拡張してカスタム・クラスにします。または、インターシステムズの既定のロジックに依存しない、完全にカスタムのセッション維持クラスを実装することもできます。このカスタマイズ方法を使用するには、%OAuth2.Server.AbstractSession を拡張して抽象メソッドを実装します。

- **[トークン生成クラス]** – %OAuth2.Server.Generate (既定値)、%OAuth2.Server.JWT、またはメソッド GenerateAccessToken() を定義するカスタム・クラスを使用します。カスタム・クラスを作成する場合は、ここに記載したクラスの 1 つをサブクラス化すると、使用するメソッドが得られるため便利です。

- **[カスタマイズ・ネームスペース]** – カスタマイズ・コードを実行するネームスペースを指定します。

- **[カスタマイズ・ロール]** – カスタマイズ・コードを実行するときに使用するロールを 1 つ以上指定します。

カスタム・サブクラスを使用する場合は、"カスタム・メソッドの実装" を参照してください。

InterSystems IRIS 承認サーバが要求を処理する方法

このセクションでは、トークンの承認コード要求または暗黙要求を受け取った InterSystems IRIS 承認サーバが実行する処理について説明します。

1. **[認証クラス]** オプションで指定されたクラスの BeforeAuthenticate() メソッドを呼び出します。このメソッドの目的は、ユーザの識別を開始する前に要求に変更を加えることです。

   既定のクラスでは、このメソッドはスタブです。

2. 次に、付与タイプが承認コードまたは暗黙付与である場合、InterSystems IRIS は以下の手順を実行します。

   a. **[認証クラス]** オプションで指定されたクラスの DisplayLogin() を呼び出します。("DirectLogin() の実装" も参照してください。)

      既定のクラスで、DisplayLogin() は単純な HTML ログイン・ページを表示します。

   b. ユーザ名が Null でない場合は、**[ユーザ検証クラス]** オプションで指定されたクラスの ValidateUser() メソッドを呼び出します。このメソッドの目的は、ユーザを検証して、(properties 配列の変更によって) トークン・エンドポイント、Userinfo エンドポイント、トークン・イントロスペクション・エンドポイントによって返されるクレームを準備することです。

      既定のクラスでは、このメソッドは例にすぎず、実稼働環境での使用には適していない可能性があります。

   c. ユーザが検証されると、**[認証クラス]** オプションで指定されたクラスの DisplayPermissions() メソッドを呼び出します。このメソッドの目的は、要求された許可の一覧を示すページをユーザに表示することです。

      既定のクラスで、このメソッドは許可の一覧を示す単純な HTML ページを表示します。

   または、付与タイプがパスワード資格情報である場合、InterSystems IRIS は **[ユーザ検証クラス]** オプションで指定されたクラスの ValidateUser() メソッドを呼び出します。

   または、付与タイプがクライアント資格情報である場合、InterSystems IRIS は **[ユーザ検証クラス]** オプションで指定されたクラスの ValidateClient() メソッドを呼び出します。

3. ユーザが許可を受け入れると、**[認証クラス]** オプションで指定されたクラスの AfterAuthenticate() メソッドを呼び出します。このメソッドの目的は、アクセス・トークンを生成する前にカスタム処理を実行することです。

   既定のクラスでは、このメソッドはスタブです。

4. **[トークン生成クラス]** オプションで指定されたクラスの GenerateAccessToken() メソッドを呼び出します。このメソッドの目的は、ユーザに返すアクセス・トークンを生成することです。

   既定のクラス (**%OAuth2.Server.Generate**) で、このメソッドは opaque 文字列のアクセス・トークンを生成します。さらに InterSystems IRIS は代替クラス (**%OAuth2.Server.JWT**) を提供し、このクラスで GenerateAccessToken() は JWT のアクセス・トークンを生成します。

既定のクラス

このセクションでは、InterSystems IRIS 承認サーバの既定のクラスについて説明します。さらに、**[トークン生成クラス]** の別のオプションとして提供される **%OAuth2.Server.JWT** クラスについても説明します。

%OAuth2.Server.Authenticate (認証クラスの既定値)

%OAuth2.Server.Authenticate クラスは、以下のメソッドを定義します (呼び出し順に記載)。

- BeforeAuthenticate() はスタブです。OK ステータスを表示して終了します。
- DisplayLogin() は、**[ログイン]** ボタンと **[キャンセル]** ボタンからなる単純なログイン・ページを作成する HTML を記述します。
- DisplayPermissions() は、要求された許可を表示する単純なページを作成する HTML を記述します。このページには、**[許可]** ボタンと **[キャンセル]** が表示されます。

- AfterAuthenticate() はスタブです。OK ステータスを表示して終了します。

%OAuth2.Server.Validate (ユーザ検証クラスの既定値)

%OAuth2.Server.Validate クラスは、[ユーザ検証クラス] オプションの既定のクラスです。

注釈　このクラスは実例で説明するためのものであり、実稼働環境での使用には適していない可能性があります。各自のニーズに応じてこのクラスを置き換えるか、サブクラス化することをお勧めします。

このクラスは、以下のサンプル・メソッドを定義します。

- ValidateUser() は以下を実行します。
  1. 指定されたユーザを IRISSYS データベースで検索します。
  2. そのユーザのパスワードを検証します。
  3. そのユーザに関する情報を含む多次元配列を取得します。
  4. この配列を使用して追加のクレームを properties オブジェクトに加えます。
- SupportedClaims() は、この承認サーバがサポートしているクレームの $LIST を返します。既定では、このメソッドは OpenID Connect Core で定義されているクレームのリストを特に返します。
- ValidateClient() (クライアント資格情報付与タイプで使用) はすべてのクライアントを受け入れて、プロパティは追加しません。

サブクラス内のこれらのメソッドはすべてオーバーライドすることができます。

OAuth2.Server.Session (セッション・クラスの既定値)

%OAuth2.Server.Session クラスは、[セッション維持クラス] オプションの既定のクラスです。このクラスは、HTTP 専用 Cookie を介してセッションを維持します。

このクラスで、GetUser() メソッドは現在のセッションにアクセスしようとします。セッションが存在する場合、このメソッドはそのセッションからユーザ名を取得して返します。セッションが存在しない場合、このメソッドはユーザ名を空の文字列で返し、出力としてエラー・ステータスも返します。

このクラスの追加情報は、クラス・リファレンスを参照してください。

%OAuth2.Server.Generate (トークン生成クラスの既定値)

%OAuth2.Server.Generate クラスは、[トークン生成クラス] オプションの既定のクラスです。このクラスは、以下のメソッドを定義します。

- GenerateAccessToken() は、opaque アクセス・トークンとしてランダムな文字列を生成します。
- IsJWT() は 0 を返します。

%OAuth2.Server.JWT (アクセス・トークン生成クラスの別のオプション)

%OAuth2.Server.JWT クラスは、[トークン生成クラス] オプションで使用できる (またはサブクラス化できる) 別のクラスです。このクラスは、以下のメソッドを定義します。

- GenerateAccessToken() は JWT を返します。承認サーバの構成の [JSON Web Token (JWT) の設定] に従って、InterSystems IRIS は JWT を返す前に JWT の署名か暗号化またはその両方を行います。
- IsJWT() は 1 を返します。

- CreateJWT() はクレームを含む JSON オブジェクトに基づいて JWT を作成し、承認サーバの構成で規定されているように JWT の署名と暗号化を行います。このメソッドは、OAuth 2.0 および OpenID Connect 使用法の仕様に準拠しており、サブクラス内でオーバーライドするべきではありません。
- AddClaims() ー 要求されたクレームを JWT に追加します。このメソッドは、以下のとおりです。

```
ClassMethod AddClaims(claims As %ArrayOfObjects,
                      properties As %OAuth2.Server.Properties,
                      json As %DynamicObject)
```

以下はその説明です。

  - claims は %OAuth2.Server.Claim インスタンスの配列です。
  - properties は、承認サーバが使用するプロパティとクレームを含む %OAuth2.Server.Properties のインスタンスです。
  - json は JWT を表すダイナミック・オブジェクトです。このオブジェクトはこのメソッドによって変更されます。

InterSystems IRIS 承認サーバのカスタム・メソッドの実装

承認サーバの動作をカスタマイズするには、"コードのカスタマイズ・オプション" の説明に従ってクラスを定義します。次に、このセクションの情報に基づいて、カスタマイズする処理手順に応じて、これらのクラスのメソッドを定義します。

1. 認証前のオプションのカスタム処理
2. ユーザの識別
3. ユーザの検証とクレームの指定
4. 必要に応じたユーザへの許可の表示
5. 認証後のオプションのカスタム処理
6. アクセス・トークンの生成

これらのサブセクションに続く最後のサブセクションでは、このサーバがクライアント資格情報付与タイプをサポートする必要がある場合にクライアントを検証する方法について説明します。クライアント資格情報付与タイプでは、前述のリストの手順 2 ～ 4 は使用しません。

認証前のオプションのカスタム処理

ここに記載されている情報は、すべての付与タイプに当てはまります。

ユーザを認証する前にカスタム処理を実行するには、認証クラスの BeforeAuthenticate() メソッドを実装します。このメソッドには、以下のシグニチャがあります。

```
ClassMethod BeforeAuthenticate(scope As %ArrayOfDataTypes,
                               properties As %OAuth2.Server.Properties) As %Status
```

以下はその説明です。

- scope は、元のクライアント要求に含まれるスコープを含む %ArrayOfDataTypes のインスタンスです。配列キーはスコープ値であり、配列値はスコープ値の対応する表示形式です。
- properties は、承認サーバが使用するプロパティとクレームを含む %OAuth2.Server.Properties のインスタンスです。"%OAuth2.Server.Properties オブジェクトの詳細" を参照してください。

必要に応じて、メソッドのこれらの引数のいずれかまたは両方を変更します。どちらの引数もユーザの識別に使用されるメソッドに後で渡されます。このメソッドは %Status を返す必要があります。

通常、このメソッドを実装する必要はありません。ただし、このメソッドの使用事例の1つとして、FHIR® で使用される launch と launch/patient のスコープを実装するのに利用するというようなものがあります。この事例では、特定の患者を含めるようにスコープを調整する必要があります。

ユーザの識別

ここに記載されている情報は、承認コード付与タイプと暗黙付与タイプにのみ当てはまります。

ユーザを識別するには、認証クラスの DisplayLogin() メソッドを実装します。DisplayLogin() メソッドには、以下のシグニチャがあります。

```
ClassMethod DisplayLogin(authorizationCode As %String,
                         scope As %ArrayOfDataTypes,
                         properties As %OAuth2.Server.Properties,
                         loginCount As %Integer = 1) As %Status
```

以下はその説明です。

- authorizationCode
- scope は、元のクライアント要求に含まれるスコープを含む %ArrayOfDataTypes のインスタンスです。このスコープは、BeforeAuthenticate() メソッドによって変更されている可能性があります。配列キーはスコープ値であり、配列値はスコープ値の対応する表示形式です。
- properties は、承認サーバが受信し、処理の早い段階にメソッドが変更したプロパティとクレームを含む %OAuth2.Server.Properties のインスタンスです。"%OAuth2.Server.Properties オブジェクトの詳細" を参照してください。
- loginCount は、ログイン試行の回数を示す整数値です。

このメソッドは、ユーザのログイン・フォームを表示する HTML を記述します。ログイン・フォームには、[ユーザ名] フィールド、[パスワード] フィールド、[承認コード] フィールド (非表示) を含める必要があります。既定の DisplayLogin() メソッドは、%CSP.Page の InsertHiddenField() メソッドを使用して、承認コードの非表示フィールドを追加します。

通常、このフォームには Login ボタンと Cancel のボタンがあります。これらのボタンによってフォームが送信されます。ユーザが Login ボタンでフォームを送信すると、このメソッドはユーザ名とパスワードを受け入れます。ユーザが Cancel ボタンでフォームを送信すると、承認プロセスは access_denied のエラーを返して終了します。

実装時に、同じページに許可を表示することも選択できます。その場合、メソッドはスコープを表示し、Accept という名前のボタンを使用してページを送信します。

このメソッドは %Status を返す必要があります。

properties.CustomProperties の更新

フォームに名前が p_ で始まる要素が含まれている場合は、その要素に特別な処理を施します。InterSystems IRIS は DisplayLogin() メソッドが返した要素の値を properties.CustomProperties 配列に追加するとき、最初に p_ 接頭語を名前から削除します。例えば、フォームに p_addme という名前の要素が含まれている場合、InterSystems IRIS は addme (および p_addme 要素の値) を properties.CustomProperties 配列に追加します。

必要に応じて properties の他のプロパティをメソッドで直接設定することもできます。

ユーザの検証とクレームの指定

ここに記載されている情報は、クライアント資格情報付与タイプを除くすべての付与タイプに当てはまります。(この付与タイプについては、"クライアントの検証" を参照してください。)

ユーザを検証し、トークン・エンドポイント、Userinfo エンドポイント、トークン・イントロスペクション・エンドポイントによって返されるクレームを指定するには、ユーザ検証クラスの ValidateUser() メソッドを定義します。このメソッドには、以下のシグニチャがあります。

```
ClassMethod ValidateUser(username As %String,
                         password As %String,
                         scope As %ArrayOfDataTypes,
                         properties As %OAuth2.Server.Properties,
                         Output sc As %Status) As %Boolean
```

以下はその説明です。

- username はユーザが提供したユーザ名です。
- password はユーザが提供したパスワードです。ユーザが既にログインしている場合、InterSystems IRIS は空の文字列の password でこのメソッドを呼び出します。この場合、メソッドは password が空の文字列であることを検知し、パスワードの確認を試みません。
- scope は、元のクライアント要求に含まれるスコープを含む %ArrayOfDataTypes のインスタンスです。このスコープは、BeforeAuthenticate() メソッドによって変更されている可能性があります。配列キーはスコープ値であり、配列値はスコープ値の対応する表示形式です。
- properties は、承認サーバが受信し、処理の早い段階にメソッドが変更したプロパティとクレームを含む %OAuth2.Server.Properties のインスタンスです。"%OAuth2.Server.Properties オブジェクトの詳細" を参照してください。
- sc はこのメソッドが設定したステータス・コードです。このコードを使用してエラーの詳細を伝えます。

このメソッドは、以下を実行します。

- パスワードが指定されたユーザ名に対応していることを確認します。
- 業務のニーズに応じて scope および properties 引数を適宜使用します。
- 必要に応じて properties オブジェクトを変更してクレーム値を指定するか、新しいクレームを追加します。以下に例を示します。

  ```
  // Setup claims for profile and email OpenID Connect scopes.
  Do properties.SetClaimValue("sub",username)
  Do properties.SetClaimValue("preferred_username",username)
  Do properties.SetClaimValue("email",email)
  Do properties.SetClaimValue("email_verified",0,"boolean")
  Do properties.SetClaimValue("name",fullname)
  ```

- エラーの場合には sc 変数を設定します。
- ユーザを有効と見なす場合は 1 を返し、それ以外の場合は 0 を返します。

ValidateUser() が値を返すと、承認サーバは properties オブジェクトの以下の値を自動的に設定します (これらの値が欠落している場合)。

- properties.ClaimValues:
  - iss — 承認サーバの URL
  - sub — client_id
  - exp — 1840 年 12 月 31 日からの秒単位による有効期限
- properties.CustomProperties:
  - client_id — 要求しているクライアントの client_id

許可の表示

ここに記載されている情報は、承認コード付与タイプと暗黙付与タイプにのみ当てはまります。

ユーザの検証後に許可を表示するには、認証クラスの DisplayPermissions() メソッドを実装します。このメソッドには、以下のシグニチャがあります。

```
ClassMethod DisplayPermissions(authorizationCode As %String,
                               scopeArray As %ArrayOfDataTypes,
                               currentScopeArray As %ArrayOfDataTypes,
                               properties As %OAuth2.Server.Properties) As %Status
```

以下はその説明です。

- authorizationCode は承認コードです。
- scopeArray は、ユーザがまだ許可を与えられていない、新たに要求されたスコープです。この引数は %ArrayOfDataTypes のインスタンスです。

  配列キーはスコープ値であり、配列値はスコープ値の対応する表示形式です。
- currentScopeArray は、ユーザが以前に許可を与えられているスコープです。この引数は %ArrayOfDataTypes のインスタンスです。

  配列キーはスコープ値であり、配列値はスコープ値の対応する表示形式です。
- properties は、承認サーバが受信し、処理の早い段階にメソッドが変更したプロパティとクレームを含む %OAuth2.Server.Properties のインスタンスです。"%OAuth2.Server.Properties オブジェクトの詳細" を参照してください。

このフォームには Accept ボタンと Cancel のボタンが必要です。これらのボタンによってフォームが送信されます。ユーザが Accept ボタンでフォームを送信すると、このメソッドは承認を続行します。ユーザが Cancel ボタンでフォームを送信すると、承認プロセスは終了します。

認証後のオプションのカスタム処理

ここに記載されている情報は、すべての付与タイプに当てはまります。

認証後にカスタム処理を実行するには、認証クラスの AfterAuthenticate() メソッドを実装します。このメソッドには、以下のシグニチャがあります。

```
ClassMethod AfterAuthenticate(scope As %ArrayOfDataTypes, properties As %OAuth2.Server.Properties) As
 %Status
```

以下はその説明です。

- scope は、承認要求が設定したスコープとこのメソッドが呼び出される前のすべての処理を含む %ArrayOfDataTypes のインスタンスです。配列キーはスコープ値であり、配列値はスコープ値の対応する表示形式です。
- properties は、承認要求が設定したプロパティおよびクレームとこのメソッドが呼び出される前のすべての処理を含む %OAuth2.Server.Properties のインスタンスです。"%OAuth2.Server.Properties オブジェクトの詳細" を参照してください。

必要に応じて、メソッドのこれらの引数のいずれかまたは両方を変更します。具体的には、プロパティを認証 HTTP 応答に追加することもできます。その場合は、プロパティを properties.ResponseProperties に追加します。

通常、このメソッドを実装する必要はありません。ただし、このメソッドの使用事例の1つとして、FHIR® で使用される launch と launch/patient のスコープを実装するのに利用するというようなものがあります。この事例では、特定の患者を含めるようにスコープを調整する必要があります。

アクセス・トークンの生成

ここに記載されている情報は、すべての付与タイプに当てはまります。

アクセス・トークンを生成するには、トークン生成クラスの GenerateAccessToken() メソッドを実装します。このメソッドには、以下のシグニチャがあります。

```
ClassMethod GenerateAccessToken(properties As %OAuth2.Server.Properties, Output sc As %Status) As
%String
```

以下はその説明です。

- properties は、承認サーバが受信し、処理の早い段階にメソッドが変更したプロパティとクレームを含む %OAuth2.Server.Properties のインスタンスです。"%OAuth2.Server.Properties オブジェクトの詳細" を参照してください。
- 出力として返される sc は、このメソッドが設定したステータス・コードです。この変数を設定してエラーの詳細を伝えます。

このメソッドはアクセス・トークンを返します。このアクセス・トークンは properties 引数に基づきます。このメソッドで、クレームを JSON 応答オブジェクトに追加することもできます。そのためには、properties オブジェクトの ResponseProperties 配列プロパティを設定します。

クライアントの検証

ここに記載されている情報は、クライアント資格情報付与タイプにのみ当てはまります。

クライアント資格情報を検証し、トークン・エンドポイント、Userinfo エンドポイント、トークン・イントロスペクション・エンドポイントによって返されるクレームを指定するには、ユーザ検証クラスの ValidateClient() メソッドを定義します。このメソッドには、以下のシグニチャがあります。

```
ClassMethod ValidateClient(clientId As %String,
                           clientSecret As %String,
                           scope As %ArrayOfDataTypes,
                           Output properties As %OAuth2.Server.Properties,
                           Output sc As %Status) As %Boolean
```

以下はその説明です。

- clientId はクライアント ID です。
- clientSecret はクライアント秘密鍵です。
- scope は、元のクライアント要求に含まれるスコープを含む %ArrayOfDataTypes のインスタンスです。このスコープは、BeforeAuthenticate() メソッドによって変更されている可能性があります。配列キーはスコープ値であり、配列値はスコープ値の対応する表示形式です。
- properties は、承認サーバが受信し、処理の早い段階にメソッドが変更したプロパティとクレームを含む %OAuth2.Server.Properties のインスタンスです。"%OAuth2.Server.Properties オブジェクトの詳細" を参照してください。
- sc はこのメソッドが設定したステータス・コードです。このコードを使用してエラーの詳細を伝えます。

このメソッドは、以下を実行します。

- クライアント秘密鍵が指定されたクライアント ID に対応していることを確認します。
- 業務のニーズに応じて scope および properties 引数を適宜使用します。
- 必要に応じて、properties オブジェクトを変更してクレーム値を指定します。以下に例を示します。

```
// Setup claims for profile and email OpenID Connect scopes.
Do properties.SetClaimValue("sub",username)
Do properties.SetClaimValue("preferred_username",username)
Do properties.SetClaimValue("email",email)
Do properties.SetClaimValue("email_verified",0,"boolean")
Do properties.SetClaimValue("name",fullname)
```

- エラーの場合には sc 変数を設定します。

- ユーザを有効と見なす場合は 1 を返し、それ以外の場合は 0 を返します。

ValidateClient() が値を返すと、承認サーバは properties オブジェクトの以下の値を自動的に設定します (これらの値が欠落している場合)。

- properties.ClaimValues:
  - iss ー 承認サーバの URL
  - sub ー client_id
  - exp ー 1840 年 12 月 31 日からの秒単位による有効期限
- properties.CustomProperties:
  - client_id ー 要求しているクライアントの client_id

%OAuth2.Server.Properties オブジェクトの詳細

前のセクションで説明したメソッドでは、properties 引数を使用していますが、この引数は%OAuth2.Server.Properties のインスタンスです。%OAuth2.Server.Properties クラスの目的は、承認サーバ・コード内のメソッド間で受け渡す必要がある情報を保持することです。このセクションでは、このクラスの基本プロパティとクレーム関連のプロパティについて説明します。このクラスには、クレームを操作するメソッドもあります。これらのメソッドについては、最後のサブセクションで説明します。

基本プロパティ

%OAuth2.Server.Properties クラスには以下の基本プロパティがあります。これらのプロパティは、カスタム・コードの内部処理の情報の伝達に使用されます。

RequestProperties

```
Property RequestProperties as array of %String (MAXLEN="");
```

承認要求のクエリ・パラメータが含まれています。

このプロパティは配列であるため、通常の配列インタフェースを使用して操作します。(これは、このクラスの他のプロパティでも同様です。)例えば、クエリ・パラメータの値を取得するには、RequestProperties.GetAt(parmname) を使用します。ここで、parmname はクエリ・パラメータの名前です。

ResponseProperties

```
Property ResponseProperties as array of %String (MAXLEN="");
```

トークン要求への JSON 応答オブジェクトに追加されるプロパティがすべて含まれます。このプロパティは必要に応じて設定します。

CustomProperties

```
Property CustomProperties as array of %String (MAXLEN="");
```

さまざまなカスタマイズ・コード間の通信に使用されるカスタム・プロパティが含まれます。"properties.CustomProperties の更新" を参照してください。

ServerProperties

```
Property ServerProperties as array of %String (MAXLEN="");
```

承認サーバがカスタマイズ・コードと共有するプロパティが含まれます。認証クラスで使用するために、クライアント・プロパティの logo_uri、client_uri、policy_uri、tos_uri がこの方法で共有されます。

クレーム関連のプロパティ

%OAuth2.Server.Properties クラスには、IntrospectionClaims、IDTokenClaims、UserinfoClaims、JWTClaims プロパティがあります。これらのプロパティは、必須のクレーム、特にカスタム・クレームに関する情報を伝達します。

このクラスには、実際のクレーム値を伝達する ClaimValues プロパティもあります。カスタマイズ・コードでクレームの値を設定する必要があります (通常は ValidateUser クラス内で)。

以下のリストで、これらのプロパティについて説明します。

IntrospectionClaims

```
Property IntrospectionClaims as array of %OAuth2.Server.Claim;
```

イントロスペクション・エンドポイントが返すクレームを指定します (必須クレームの基本セットの範囲を超えて指定)。承認サーバは scope、client_id、username、token_type、exp、iat、nbf、sub、aud、iss、jti クレームを (このプロパティに含まれていない場合でも) 返します。

ほとんどの場合、このプロパティの値には空の文字列を使用できます。このプロパティは、claims 要求パラメータをサポートするために指定されます (詳細は、OpenID Connect Core のセクション 5.5 を参照してください)。

形式的にはこのプロパティは配列であり、その配列キーは (ClaimValues プロパティの名前と一致する) クレーム名であり、その配列値は %OAuth2.Server.Claim のインスタンスです。%OAuth2.Server.Claim クラスには以下のプロパティがあります。

- Essential

  ```
  property Essential as %Boolean [ InitialExpression = 0 ];
  ```

  クレームが必須、任意選択のいずれであるかを指定します。値 1 は必須を意味し、値 0 は任意選択を意味します。

- Values

  ```
  property Values as list of %String(MAXLEN=2048);
  ```

  このクレームの許容値のリストを指定します。

通常、クレームの値は ValidateUser クラスで設定されます。

IDTokenClaims

```
Property IDTokenClaims as array of %OAuth2.Server.Claim;
```

承認サーバが IDToken で必要とするクレームを指定します (必須クレームの基本セットの範囲を超えて指定)。承認サーバは、iss、sub、exp、aud、azp クレームを (このプロパティに含まれていない場合でも) 必要とします。

このプロパティはオブジェクトの配列です。詳細は、IntrospectionClaims プロパティの項目を参照してください。

ほとんどの場合、このプロパティの値には空の文字列を使用できます。このプロパティは、claims 要求パラメータをサポートするために指定されます (詳細は、OpenID Connect Core のセクション 5.5 を参照してください)。

UserinfoClaims

```
Property UserinfoClaims as array of %OAuth2.Server.Claim;
```

Userinfo エンドポイントが返すクレームを指定します (必須クレームの基本セットの範囲を超えて指定)。承認サーバは sub クレームを (このプロパティに含まれていない場合でも) 返します。

ほとんどの場合、このプロパティの値には空の文字列を使用できます。このプロパティは、OpenID Connect Core のセクション 5.5 をサポートするために提供されます。

このプロパティはオブジェクトの配列です。詳細は、IntrospectionClaims プロパティの項目を参照してください。

クレームは、スコープと要求のクレーム・パラメータに基づいて定義されます。クレームの戻り値は、ClaimValues プロパティの同じキーを持ちます。通常、クレームの値は ValidateUser クラスで設定されます。

JWTClaims

```
Property JWTClaims as array of %OAuth2.Server.Claim;
```

JWT ベースの既定のアクセス・トークン・クラス (%OAuth2.Server.JWT) が返す JWT アクセス・トークンで必要とされるクレームを必須クレームの基本セットの範囲を超えて指定します。承認サーバは、iss、sub、exp、aud、jti クレームを (このプロパティに含まれていない場合でも) 返します。

このプロパティはオブジェクトの配列です。詳細は、IntrospectionClaims プロパティの項目を参照してください。

このクレームはカスタマイズ・コードによって定義されます。通常、クレームの値は ValidateUser クラスで設定されます。

ClaimValues

```
property ClaimValues as array of %String(MAXLEN=1024);
```

実際のクレーム値とその型を指定します。このプロパティを操作するには、次のセクションのメソッドを使用します。

このプロパティを直接操作する必要がある場合は、このプロパティが以下の特徴を持つ配列であることに注意してください。

- 配列キーはクレーム名です。
- 配列値は $LISTBUILD(type,value) の形式をとります。ここで、type は値のタイプを保持し、value は実際の値を保持します。type には "string"、"boolean"、"number"、または "object" を指定できます。type が "object" の場合、value は文字列としてシリアル化された JSON オブジェクトとなります。

  value は $LIST 構造にすることもできます。この場合、クレーム値はシリアル化されるときに JSON 配列としてシリアル化され、それぞれの配列項目は指定された type となります。

クレームを操作するメソッド

%OAuth2.Server.Properties クラスは、ClaimValues プロパティの操作を簡素化するために使用できるインスタンス・メソッドも提供します。

SetClaimValue()

```
Method SetClaimValue(name As %String, value As %String, type As %String = "string")
```

name 引数で指定したクレームの値を設定することで、ClaimValues プロパティを更新します。type 引数はクレームの型を示します。型には "string" (既定値)、"boolean"、"number"、"object" があります。type が "object" の場合、value は文字列としてシリアル化された JSON オブジェクトでなければなりません。

value は $LIST 構造にすることもできます。この場合、クレーム値はシリアル化されるときに JSON 配列としてシリアル化され、それぞれの配列項目は指定された type となります。

RemoveClaimValue()

```
Method RemoveClaimValue(name As %String)
```

name 引数で指定したクレームを削除することで、ClaimValues プロパティを更新します。

GetClaimValue()

```
Method GetClaimValue(name As %String, output type) As %String
```

ClaimValues プロパティを調べて、name 引数で指定されたクレームの値を返します。出力として返される type 引数はクレームの型を示します。SetClaimValue() を参照してください。

NextClaimValue()

```
Method NextClaimValue(name As %String) As %String
```

指定されたクレームの後にある (ClaimValues プロパティの) 次のクレームの名前を返します。

承認サーバ・エンドポイントの場所

OAuth 2.0 承認サーバとして InterSystems IRIS インスタンスを使用する場合、承認エンドポイントの URL は次のようになります。

エンドポイント	URL
発行者エンドポイント	`https://serveraddress/oauth2`
承認エンドポイント	`https://serveraddress/oauth2/authorize`
トークン・エンドポイント	`https://serveraddress/oauth2/token`
Userinfo エンドポイント	`https://serveraddress/oauth2/userinfo`
トークン・イントロスペクション・エンドポイント	`https://serveraddress/oauth2/introspection`
トークン取り消しエンドポイント	`https://serveraddress/oauth2/revocation`

いずれの場合も、serveraddress は InterSystems IRIS インスタンスを実行しているサーバの IP アドレスまたはホスト名になります。

InterSystems IRIS OAuth 2.0 承認サーバでのクライアント定義の作成

このセクションでは、クライアントを動的に登録していない場合に、InterSystems IRIS OAuth 2.0 承認サーバでクライアント定義を作成する方法について説明します。まず、このページの前述の説明に従って InterSystems IRIS OAuth 2.0 承認サーバを設定します。管理ポータルで以下の手順を実行します。

1. **[システム管理]**→**[セキュリティ]**→**[OAuth 2.0]**→**[サーバ構成]** を選択します。
2. **[クライアント構成]** ボタンをクリックしてクライアント記述を表示します。このテーブルは最初は空です。
3. **[一般]** タブで、以下の詳細を指定します。
   - **[名前]** – このクライアントの一意の名前を指定します。
   - **[説明]** – 任意で説明を指定します。
   - **[クライアントの種別]** – このクライアントのタイプを指定します。選択項目には **[パブリック]** (パブリック・クライアント、RFC 6749 に準拠)、**[機密]** (機密クライアント、RFC 6749 に準拠)、**[リソース]** (専用のリソース・サーバ) があります。

- **[リダイレクト URL]** – このクライアントで予期される 1 つ以上のリダイレクト URL。
- **[サポートされている付与タイプ]** – このクライアントがアクセス・トークンを作成するために使用できる付与タイプを指定します。少なくとも 1 つ選択します。
- **[サポートされるレスポンス・タイプ]** – クライアントが限定使用するOAuth 2.0 response_type値を選択します。
- **[認証タイプ]** – 承認サーバへの HTTP 要求で使用する認証タイプを選択します (RFC 6749 または OpenID Connect Core のセクション 9 で規定)。以下のいずれかを選択します。
  - **なし**
  - **基本**
  - **フォームエンコードされた本文**
  - **クライアント秘密鍵 JWT**
  - **秘密鍵 JWT**
- **[認証署名アルゴリズム]** – トークン・エンドポイントでこのクライアントを認証するために使用される JWT の署名に必要なアルゴリズムを選択します ([認証タイプ] が **[クライアント秘密鍵 JWT]** または **[秘密鍵 JWT]** である場合)。オプションを選択しない場合、OpenID プロバイダおよび Relying Party でサポートされる任意のアルゴリズムが使用されます。

4. 必要な場合は、**[クライアント資格情報]** タブを選択し、次の詳細を表示します。
   - **[クライアント ID]** – RFC 6749 で規定されているクライアント ID。InterSystems IRIS がこの文字列を生成します。
   - **[クライアント秘密鍵]** – RFC 6749 で規定されているクライアント秘密鍵。InterSystems IRIS がこの文字列を生成します。
5. **[クライアント情報]** タブで、以下の詳細を指定します。
   - **[起動 URL]** – このクライアントの起動に使用する URL を指定します。状況によっては、この値をクライアントの識別に使用することも、aud クレームの値として使用することもできます。
   - **[承認の表示]** セクション:
     - **[クライアント名]** – エンド・ユーザに表示するクライアントの名前を指定します。
     - **[ロゴ URL]** – クライアント・アプリケーションのロゴを参照する URL を指定します。このオプションを指定すると、承認サーバは承認中にこのイメージをエンド・ユーザに表示します。このフィールドの値は、有効なイメージ・ファイルを指している必要があります。
     - **[クライアント・ホーム・ページ]** – クライアントのホーム・ページの URL を指定します。このフィールドの値は、有効な Web ページを指している必要があります。このオプションを指定すると、承認サーバは、この URL を分かりやすい形でエンド・ユーザに示します。
     - **[ポリシー URL]** – プロファイル・データの取り扱い方法を確認できるよう Relying Party Client エンド・ユーザに提供する URL を指定します。このフィールドの値は、有効な Web ページを指している必要があります。
     - **[サービス条件の URL]** – Relying Party のサービス条件を確認できるように Relying Party Client がエンド・ユーザに提供する URL を指定します。このフィールドの値は、有効な Web ページを指している必要があります。
   - **[連絡先電子メール]** – クライアント・アプリケーションの責任者への連絡に使用する適切な電子メール・アドレスをコンマで区切ったリストです。
   - **[既定の最長経過時間]** – 既定の最長認証経過時間 (秒単位) を指定します。このオプションを指定した場合、最長認証経過時間に達したとき、エンド・ユーザをアクティブに再認証する必要があります。max_age 要求パラ

メータは、この既定値をオーバーライドします。このオプションを省略した場合に、既定で最長認証経過時間は設定されません。

- **[既定のスコープ]** ― アクセス・トークン要求の既定のスコープを空白で区切ったリストで指定します。

6. **[JWT 設定]** タブで、以下の詳細を指定します。
   - **[JSON Web Token (JWT) の設定]** ― 承認サーバから JWT のシグニチャを検証し、承認サーバに送信される JWT を暗号化するためにクライアントが使用する公開鍵のソースを指定します。

     既定では、ダイナミック登録プロセスは JWKS (JSON Web Key Set) のペアを生成します。一方の JWKS は秘密鍵で、(アルゴリズムあたり) 必要なすべての秘密鍵だけでなく、対称鍵として使用するクライアント秘密鍵も含んでいます。この JWKS は共有されません。他方の JWKS には、対応する公開鍵が含まれていて、公開されて利用できます。ダイナミック登録プロセスは、公開の JWKS のクライアントへのコピーも実行します。

     その他のオプションは以下のとおりです。

     - **[URL から JWKS]** ― 公開の JWKS を指す URL を指定した後、InterSystems IRIS に JWKS をロードします。
     - **[ファイルから JWKS]** ― 公開の JWKS を含むファイルを選択し、そのファイルを InterSystems IRIS にロードします。
     - **[X509 証明書]** ― 詳細は、"証明書と JWT (JSON Web Token)" の "OAuth 2.0 承認サーバの証明書の使用" を参照してください。

     これらのオプションのいずれかにアクセスするには、まず **[ダイナミック登録以外のソース]** を選択します。

7. **[保存]** を選択します。

このタスクを実行するには、**%Admin_OAuth2_Registration** リソースの USE 権限を持つユーザとしてログインする必要があります。

JWT で使用するキーの回転

ほとんどの場合、新しい公開/秘密鍵ペアを承認サーバに生成させることができます。これは、非対称の RS256、RS384、および RS512 の各アルゴリズムに使用する RSA 鍵にのみ適用されます。(例外は、**[X509 証明書]** として、**[ダイナミック登録以外のソース]** を指定する場合です。この場合、新しいキーは生成できません。)

新しい公開/秘密鍵ペアの生成は、キーの回転と呼ばれています。このプロセスは、新しい秘密 RSA 鍵および関連する公開 RSA 鍵を秘密と公開の JWKS に追加します。

承認サーバでキーの回転を実行すると、承認サーバは新しい秘密 RSA 鍵を使用して、クライアントに送信する JWT に署名します。同様に、承認サーバは新しい公開 RSA 鍵を使用して、クライアントに送信する JWT を暗号化します。承認サーバは、クライアントから受信した JWT を解読するために新しい RSA 鍵を使用しますが、これが失敗したら古い RSA 鍵を使用するため、古い公開 RSA 鍵を使用して作成された JWT を解読できます。

最後に、クライアントから受信した署名済み JWT を承認サーバが検証できない場合、承認サーバにクライアントの公開の JWKS の URL があると、承認サーバは新しい公開の JWKS を取得し、署名の検証を再試行します。(ダイナミック Discovery を使用した場合や、構成で **[URL から JWKS]** オプションが指定された場合、承認サーバはクライアントの公開の JWKS の URL を持っています。それ以外の場合には承認サーバはこの URL を持っていません。)

承認サーバのキーを回転するには、以下の手順を実行します。

1. **[システム管理]**→**[セキュリティ]**→**[OAuth 2.0]**→**[サーバ構成]** を選択します。

   承認サーバの構成が表示されます。

2. **[キーの回転]** ボタンを選択します。

注釈　対称の HS256、HS384、および HS512 の各アルゴリズムは、常に対称鍵としてクライアントの秘密鍵を使用します。

承認サーバのキーの回転の API

承認サーバでキーをプログラムによって回転させるには、**OAuth2.Server.Configuration** の RotateKeys() メソッドを呼び出します。

新しいクライアントの JWKS を取得するには、**OAuth2.Server.Client** の UpdateJWKS() メソッドを呼び出します。

これらのメソッドの詳細は、クラス・リファレンスを参照してください。

クライアントからの新しい公開の JWKS の取得

ほとんどの場合、クライアントは JWKS の公開/秘密鍵ペアを生成します。公開の JWKS を承認サーバが受信する方法はさまざまです。1 つの方法は、公開の JWKS をクライアントが URL で指定する方法です。"InterSystems IRIS OAuth 2.0 承認サーバでのクライアント定義の作成" の **[URL から JWKS]** オプションを参照してください。

クライアントが **[URL から JWKS]** で定義されていて、クライアントが JWKS の新しいペアを生成する場合、同じ URL から新しい公開の JWKS を承認サーバに取得させることができます。そのためには、以下の操作を実行します。

1. 管理ポータルで **[システム管理]**→**[セキュリティ]**→**[OAuth 2.0]**→**[サーバ構成]** を選択します。

   承認サーバの構成が表示されます。

2. **[JWKS の更新]** ボタンを選択します。

クライアントが **[URL から JWKS]** で定義されておらず、クライアントが JWKS の新しいペアを生成する場合、公開の JWKS を取得し、これを承認サーバに送信し、ファイルからロードする必要があります。

プログラムによる構成項目の作成方法

管理ポータルを使用して OAuth 2.0 のクライアント、リソース・サーバ、承認サーバを構成する方法については他の記事で説明しています。これらの構成をプログラムで作成することもできます。以下のサブセクションでは、クライアント (リソース・サーバを含む) と承認サーバの作成方法について詳しく説明します。

プログラムによるクライアント構成項目の作成方法

OAuth 2.0 クライアントまたは OAuth 2.0 リソース・サーバの構成項目をプログラムで作成するには、以下の手順を実行します。

1. サーバ記述を作成します。
2. 関連付けられたクライアント構成を作成します。

サーバ記述の作成

サーバ記述は、**OAuth2.ServerDefinition** のインスタンスです。サーバ記述を作成するには次の手順に従います。

1. **%SYS** ネームスペースに切り替えます。
2. 承認サーバが Discovery をサポートする場合は、**%SYS.OAuth2.Registration** の Discover() メソッドを呼び出します。このメソッドは、以下のとおりです。

```
ClassMethod Discover(issuerEndpoint As %String,
                     sslConfiguration As %String,
                     Output server As OAuth2.ServerDefinition) As %Status
```

   以下はその説明です。

   - issuerEndpoint は、承認サーバの識別に使用するエンドポイントの URL を指定します。

- sslConfiguration は、Discover() メソッドの呼び出しに使用する InterSystems IRIS SSL/TLS 構成のエイリアスを指定します。
- 出力として返される server は、OAuth2.ServerDefinition のインスタンスです。

3. 次に、返された OAuth2.ServerDefinition インスタンスを保存します。

また、承認サーバが Discovery をサポートしない場合、次の手順を実行します。

1. %SYS ネームスペースに切り替えます。
2. OAuth2.ServerDefinition のインスタンスを作成します。
3. インスタンスのプロパティを設定します。ほとんどの場合、プロパティの名前は管理ポータルに表示されるラベルと一致します (スペースと大文字化を除く)。詳細は、"サーバ記述の手動による作成" を参照してください。これらのプロパティは、以下のとおりです。
   - IssuerEndpoint
   - SSLConfiguration
   - InitialAccessToken。[**登録アクセストークン**] フィールドに対応します。
   - Metadata。OAuth2.Server.Metadata のインスタンスで、多くのプロパティが含まれます。OpenID Provider Metadata (https://openid.net/specs/openid-connect-discovery-1_0.html) を参照してください。

   ServerCredentials の詳細は、"証明書と JWT (JSON Web Token)" の "OAuth 2.0 承認サーバの証明書の使用" を参照してください。
4. インスタンスを保存します。

クライアント構成の生成

クライアント構成は、OAuth2.Client のインスタンスです。クライアント構成を作成するには、次の手順を実行します。

1. %SYS ネームスペースに切り替えます。
2. OAuth2.Client のインスタンスを作成します。
3. インスタンスのプロパティを設定します。ほとんどの場合、プロパティの名前は管理ポータルに表示されるラベルと一致します (スペースと大文字化を除く)。詳細は、"クライアントの構成および動的な登録" を参照してください。これらのプロパティは、以下のとおりです。
   - ApplicationName
   - ClientId。クライアントを動的に登録する場合、手動で設定する必要はありません。
   - ClientSecret。クライアントを動的に登録する場合、手動で設定する必要はありません。
   - DefaultScope
   - Description
   - Enabled
   - JWTInterval
   - Metadata。OAuth2.Client.Metadata のインスタンスで、多くのプロパティが含まれます。詳細は、Client Metadata (http://openid.net/specs/openid-connect-registration-1_0-19.html) を参照してください。
   - RedirectionEndpoint。[**応答を受信するために承認サーバに対して指定されるクライアント URL**] オプションに対応します。このプロパティのタイプは %OAuth2.Endpoint です。OAuth2.Endpoint クラスは、UseSSL、Host、Port、Prefix のプロパティを持つシリアル・クラスです。

- SSLConfiguration
- ServerDefinition。以前に作成した OAuth2.ServerDefinition のインスタンスであることが必要です。

ClientCredentials の詳細は、"証明書と JWT (JSON Web Token)" の "OAuth 2.0 クライアントの証明書の使用" を参照してください。

4. 承認サーバがダイナミック・クライアント登録をサポートする場合は、%SYS.OAuth2.Registration の RegisterClient() メソッドを呼び出します。このメソッドは、以下のとおりです。

```
ClassMethod RegisterClient(applicationName As %String) As %Status
```

applicationName はクライアント・アプリケーションの名前です。

このメソッドはクライアントを登録し、(クライアント ID およびクライアント秘密鍵を含む) クライアント・メタデータを取得した後、OAuth2.Client のインスタンスを更新します。

プログラムによるサーバ構成項目の作成方法

OAuth 2.0 承認サーバの構成項目をプログラムで作成するには、以下の手順を実行します。

1. 承認サーバ構成を作成します。

   1 つの InterSystems IRIS インスタンス上に複数の承認サーバ構成を定義できないことに注意してください。また、この構成を作成するには、**%Admin_OAuth2_Server** リソースの USE 権限を持つユーザとしてログインする必要があります。

2. 関連付けられたクライアント記述を作成します。

承認サーバ構成の作成

認証サーバ構成は、OAuth2.Server.Configuration のインスタンスです。承認サーバ構成を作成するには、次の手順を実行します。

1. %SYS ネームスペースに切り替えます。
2. OAuth2.Server.Configuration のインスタンスを作成します。
3. インスタンスのプロパティを設定します。ほとんどの場合、プロパティの名前は管理ポータルに表示されるラベルと一致します (スペースと大文字化を除く)。詳細は、"承認サーバの構成" を参照してください。これらのプロパティは、以下のとおりです。
   - AccessTokenInterval
   - AllowUnsupportedScope
   - AudRequired。[対象者は必須] オプションに対応します
   - AuthenticateClass
   - AuthorizationCodeInterval
   - ClientSecretInterval
   - CustomizationNamespace
   - CustomizationRoles
   - DefaultScope
   - Description
   - EncryptionAlgorithm

- GenerateTokenClass
- IssuerEndpoint。[**発行者エンドポイント**] オプションに対応します。OAuth2.Endpoint タイプです。OAuth2.Endpoint クラスは、UseSSL、Host、Port、Prefix のプロパティを持つシリアル・クラスです。
- JWKSFromCredentials
- KeyAlgorithm
- Metadata。OAuth2.Server.Metadata のインスタンスで、多くのプロパティが含まれます。OpenID Provider Metadata (https://openid.net/specs/openid-connect-discovery-1_0.html) を参照してください。
- RefreshTokenInterval
- ReturnRefreshToken
- SSLConfiguration
- SessionClass
- SessionInterval。[**セッション終了間隔**] オプションに対応します
- SigningAlgorithm
- SupportSession。[**ユーザ・セッションのサポート**] オプションに対応します
- SupportedScopes。[**スコープ**] と [**説明**] の列を持つテーブルに対応します。このプロパティは文字列の配列であるため、SetAt()、GetAt() などの通常の配列インタフェースを使用します。
- ValidateUserClass

署名、キー管理、暗号化のアルゴリズムに使用できる値については、%OAuth2.JWT のクラス・リファレンスを参照してください。

ServerCredentials および ServerPassword の詳細は、"証明書と JWT (JSON Web Token)" の "OAuth 2.0 承認サーバの証明書の使用" を参照してください。

4. `OAuth2.Server.Configuration.Save()` メソッドを使用してインスタンスを保存します。`%Save()` メソッドではなく `Save()` メソッドを使用する必要があります。このメソッドには、Web アプリケーションの作成などの追加の機能が用意されているからです。

InterSystems IRIS では、このクラスの複数インスタンスの使用はサポートされていません。

また、このインスタンスを保存するには、**`%Admin_OAuth2_Server`** リソースの USE 権限を持つユーザとしてログインする必要があります。

クライアント記述の作成

クライアント記述は、OAuth2.Server.Client のインスタンスです。クライアント記述を作成するには、次の手順を実行します。

1. %SYS ネームスペースに切り替えます。
2. OAuth2.Server.Client のインスタンスを作成します。
3. インスタンスのプロパティを設定します。ほとんどの場合、プロパティの名前は管理ポータルに表示されるラベルと一致します (スペースと大文字化を除く)。詳細は、"クライアント記述の作成" を参照してください。これらのプロパティは、以下のとおりです。
   - ClientCredentials
   - ClientType
   - DefaultScope

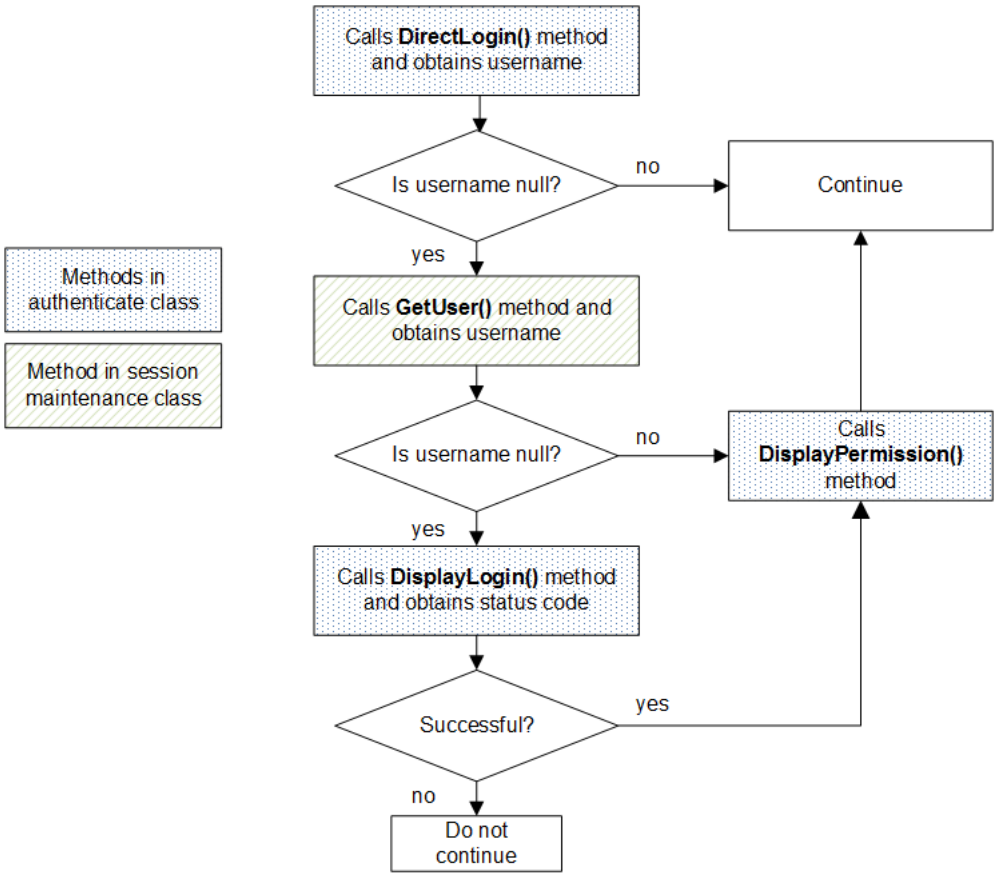
- Description
- LaunchURL
- Metadata。OAuth2.Client.Metadata のインスタンスで、多くのプロパティが含まれます。詳細は、Client Metadata (http://openid.net/specs/openid-connect-registration-1_0-19.html) を参照してください。
- Name
- RedirectURL。[リダイレクト URL] オプションに対応します。このプロパティは文字列の配列であるため、SetAt()、GetAt() などの通常の配列インタフェースを使用します。

4. インスタンスを保存します。

   システムは、ClientId および ClientSecret プロパティの値を生成します。

DirectLogin() の実装

OAuth 2.0 承認サーバとして InterSystems IRIS® を使用するときは、通常、認証クラスの DisplayLogin() メソッドを実装します。このメソッドは、ユーザがユーザ名とパスワードを入力してログインするページを表示します。サーバによる認証時にログイン・フォームを表示せず、現在のセッションも使用されないようにする場合は、認証クラスの DirectLogin() メソッドを実装します。以下のフローチャートは、アクセス・トークンの要求を処理するときに InterSystems IRIS 承認サーバがユーザを識別する仕組みを示しています。



既定では、GetUser() メソッドは、前のログインで入力されたユーザ名を取得します。

DirectLogin() を実装すると、DisplayPermissions() は呼び出されません。DirectLogin() は許可を表示する役割を果たすからです。

DirectLogin() メソッドには、以下のシグニチャがあります。

```
ClassMethod DirectLogin(scope As %ArrayOfDataTypes,
                        properties As %OAuth2.Server.Properties,
                        Output username As %String,
                        Output password As %String) As %Status
```

以下はその説明です。

- scope は、元のクライアント要求に含まれるスコープを含む %ArrayOfDataTypes のインスタンスです。このスコープは、BeforeAuthenticate() メソッドによって変更されている可能性があります。配列キーはスコープ値であり、配列値はスコープ値の対応する表示形式です。
- properties は、承認サーバが受信し、処理の早い段階にメソッドが変更したプロパティとクレームを含む %OAuth2.Server.Properties のインスタンスです。"%OAuth2.Server.Properties オブジェクトの詳細" を参照してください。
- 出力として返される username はユーザ名です。
- 出力として返される password は対応するパスワードです。

実装では、独自のロジックを使用して username および password 引数を設定します。そのためには、必要に応じて scope および properties 引数を使用します。アクセスを拒否するには、メソッドで username 引数を $char(0) に設定します。この場合、承認サーバは `access_denied` のエラーを返します。

メソッドで properties のプロパティを設定することもできます。このオブジェクトは後続の処理で利用できます。

このメソッドは %Status を返す必要があります。

証明書と JWT (JSON Web Token)

OAuth 2.0 のそれぞれのパーティが公開/秘密鍵ペアを持っている必要があります。これらの鍵ペアに対して、証明書およびその秘密鍵を使用できます。ただし、これは一般的な手法ではありません。このページでは、次の各シナリオの詳細について説明します。

- OAuth 2.0 クライアント
- OAuth 2.0 リソース・サーバ
- OAuth 2.0 承認サーバ

いずれの場合でも、秘密鍵と対応する証明書の生成では、インターシステムズ公開鍵インフラストラクチャを使用できます。

注釈　InterSystems IRIS は JWKS (JSON Web Key Set) のペアを生成できます。一方の JWKS は秘密鍵で、(アルゴリズムあたり) 必要なすべての秘密鍵だけでなく、対称鍵として使用するクライアント秘密鍵も含んでいます。この JWKS は共有されません。他方の JWKS には、対応する公開鍵が含まれていて、公開されて利用できます。JWKS を生成するオプションを使用する場合は、このページは無視してください。

OAuth 2.0 クライアントの証明書の使用

OAuth 2.0 クライアントは承認サーバから (暗号化や署名を施した) JWT を受信できます。同様に、このクライアントは承認サーバへ (暗号化や署名を施した) JWT を送信することができます。これらの目的で証明書/秘密鍵のペアを使用する場合、以下の表を参考にして必要になる証明書を特定してください。

シナリオ	クライアント構成の要件
次のいずれか： ・ クライアントは、承認サーバが送信した JWT のシグニチャを検証する必要がある ・ クライアントは、承認サーバへ送信する JWT を暗号化する必要がある	承認サーバが所有する証明書と、サーバ証明書に署名する CA（認証機関）証明書を取得します。この証明書の公開鍵は、シグニチャ検証と暗号化で使用されます。
次のいずれか： ・ クライアントは、JWT に署名してから承認サーバへ送信する必要がある ・ クライアントは、承認サーバが送信した JWT を復号化する必要がある	クライアント用の秘密鍵を取得し、対応する証明書とその証明書に署名する CA 証明書を取得します。秘密鍵は署名と復号化に使用します。

いずれの場合も、そのクライアント Web アプリケーションを含む同じインスタンスで次の手順も実行する必要があります。

- InterSystems IRIS が使用する信頼できる証明書を用意します。信頼された証明書には、クライアントの証明書に署名する証明書と承認サーバの証明書を組み込む必要があります（必要とする証明書に応じていずれかまたは両方）。
- InterSystems IRIS で証明書を使用できるようにするための InterSystems IRIS 資格情報セットを作成します。

  クライアント証明書の場合は、資格情報セットを作成するときに秘密鍵を読み込み、秘密鍵のパスワードを入力します。
- クライアントを構成する場合は、[X509 資格情報から JWT 設定を作成] オプションを選択します。また、以下の指定を行います。
  - [X.509 証明書] ― クライアントの証明書を使用し対応秘密鍵（`ClientConfig` など）を含む資格情報セットを選択します。
  - [秘密鍵パスワード] ― この証明書の秘密鍵のパスワードを入力します。

OAuth 2.0 リソース・サーバの証明書の使用

OAuth 2.0 リソース・サーバは、承認サーバから（暗号化や署名を施した）JWT を受信できます。同様に、リソース・サーバは承認サーバへ（暗号化や署名を施した）JWT を送信することができます。これらの目的で証明書/秘密鍵のペアを使用する場合、以下の表を参考にして必要になる証明書を特定してください。

シナリオ	リソース・サーバ構成の要件
リソース・サーバは、承認サーバが送信した JWT のシグニチャを検証する必要がある	承認サーバが所有する証明書と、サーバ証明書に署名する CA 証明書を取得します。この証明書の公開鍵は、シグニチャ検証と暗号化で使用されます。
リソース・サーバは、承認サーバへ送信する JWT を暗号化する必要がある	
リソース・サーバは、JWT に署名してから承認サーバへ送信する必要がある	リソース・サーバ用の秘密鍵を取得し、対応する証明書とその証明書に署名する CA 証明書を取得します。秘密鍵は署名と復号化に使用します。
リソース・サーバは、承認サーバが送信した JWT を復号化する必要がある	

いずれの場合も、そのリソース・サーバ Web アプリケーションを含む同じインスタンスで次の手順も実行する必要があります。

- InterSystems IRIS が使用する信頼できる証明書を用意します。信頼された証明書には、リソース・サーバの証明書に署名する証明書と承認サーバの証明書を組み込む必要があります (必要とする証明書に応じていずれかまたは両方)。
- InterSystems IRIS で証明書を使用できるようにするための InterSystems IRIS 資格情報セットを作成します。

  リソース・サーバの証明書では、資格情報セットを作成するときに秘密鍵を読み込み、秘密鍵のパスワードを入力します。
- リソース・サーバを構成する場合は、[**X509 資格情報から JWT 設定を作成**] オプションを選択します。また、以下の指定を行います。
  - [**X.509 証明書**] – リソース・サーバの証明書を使用し対応秘密鍵 (`ResourceConfig` など) を含む資格情報セットを選択します。
  - [**秘密鍵パスワード**] – この証明書の秘密鍵のパスワードを入力します。

OAuth 2.0 承認サーバの証明書の使用

OAuth 2.0 承認サーバは、そのクライアントから (暗号化や署名を施した) JWT を受信できます。同様に、承認サーバはそのクライアントへ (暗号化や署名を施した) JWT を送信することができます。これらの目的で証明書/秘密鍵のペアを使用する場合、以下の表を参考にして必要になる証明書を特定してください。

シナリオ	承認サーバ構成の要件
承認サーバは、クライアントが送信した JWT のシグニチャを検証する必要がある	そのクライアントが所有する証明書と、証明書に署名する CA 証明書を取得します。この証明書の公開鍵は、シグニチャ検証と暗号化で使用されます。
承認サーバは、クライアントへ送信する JWT を暗号化する必要がある	
承認サーバは、JWT に署名してからクライアントへ送信する必要がある	承認サーバ用の秘密鍵を取得し、対応する証明書とその証明書に署名する CA 証明書を取得します。秘密鍵は署名と復号化に使用します。
承認サーバは、クライアントが送信した JWT を復号化する必要がある	

いずれの場合も、その承認サーバを含む同じインスタンスで次の手順も実行する必要があります。

- InterSystems IRIS が使用する信頼できる証明書を用意します。信頼された証明書には、クライアントの証明書に署名する証明書と承認サーバの証明書を組み込む必要があります (必要とする証明書に応じていずれかまたは両方)。
- InterSystems IRIS で証明書を使用できるようにするための InterSystems IRIS 資格情報セットを作成します。

  承認サーバの証明書では、資格情報セットを作成するときに秘密鍵を読み込み、秘密鍵のパスワードを入力します。
- サーバを構成する場合、[**JWT 設定**] タブを選択します。このタブで、[**X509 資格情報から JWT 設定を作成**] オプションを選択します。また、以下の指定を行います。
  - [**X509 証明書**] – 承認サーバの証明書を使用し対応秘密鍵 (`AuthConfig` など) を含む資格情報セットを選択します。
  - [**秘密鍵パスワード**] – この証明書の秘密鍵のパスワードを入力します。

- サーバ上でクライアント定義を作成する場合、[JWT 設定] タブを選択します。このタブで、[ダイナミック登録以外のソース] では、[X509 証明書] を選択します。また、[クライアント資格情報] では、クライアント証明書を使用する認証情報セット (ClientConfig など)を選択します。

JWT ヘッダの操作

ここでは、JWT ヘッダをカスタマイズする方法と、JWT ヘッダ内のカスタム値を処理する方法を説明します。JWT は、承認サーバ、または OAuth フレームワーク外部のカスタム・コードによって生成されます。

ヘッダ値の追加 (承認サーバ)

承認サーバによって JWT トークンが生成される場合、使用される署名アルゴリズムと暗号化アルゴリズムに基づいて、alg、enc、kid、typ、および cty の各ヘッダが自動的に生成されます。これらのヘッダをカスタム・コードで直接操作することはできません。ただし、他のヘッダ値は JWTHeaderClaims 配列を使用して JWT ヘッダに追加できます。例えば、JWTHeaderClaims を使用して、jku および jwk のヘッダ・パラメータをトークンに追加できます。jku の場合、関連する jwk_uri または JWKS がサーバによって自動的に JOSE 配列に追加されます。RFC 7515 で定義されているヘッダ・パラメータの一部はサポートされないことに注意してください。JWTHeaderClaims 配列を使用して、任意のカスタム値を JWT ヘッダに追加することもできます。

例えば、以下のコードを %OAuth2.Server.Validate のサブクラスに追加すると、2 つの標準ヘッダと 1 つのカスタム・ヘッダ値を JWT ヘッダに追加できます。

```
ClassMethod ValidateUser(username As %String, password As %String, scope As %ArrayOfDataTypes, properties
 As %OAuth2.Server.Properties, Output sc As %Status) As %Boolean
{
  ...
  Do properties.SetClaimValue("co","intersystems")

  Do properties.JWTHeaderClaims.SetAt("","jku")
  Do properties.JWTHeaderClaims.SetAt("","co")
  Do properties.JWTHeaderClaims.SetAt("","iss")
  ...
}
```

ヘッダ値の追加 (JWT の直接生成)

ObjectToJWT() メソッドを使用して、カスタム・コードによって OAuth フレームワークの外部で JWT を生成できます。このメソッドは、JOSE ヘッダを表す文字列の配列を最初のパラメータとして受け入れます。JWT ヘッダに値を追加するには、ObjectToJWT メソッドを呼び出す前に JOSE 配列に値を追加するだけです。例えば、jku ヘッダ・パラメータを JOSE ヘッダに追加するには、以下のように指定します。

```
Set JOSE("jku")=""
Set JOSE("jku_local")=%server.Metadata."jwks_uri"
Set JOSE("jku_remote")=%client.Metadata."jwks_uri"
// set JOSE("sigalg") and/or JOSE("encalg") and JOSE("keyalg")
Set sc=##class(%OAuth2.JWT).ObjectToJWT(.JOSE,json,%server.PrivateJWKS,%client.RemotePublicJWKS,.JWT)
```

標準のヘッダ・パラメータに対応する JOSE 配列ノードの詳細は、クラス・リファレンスを参照してください。

カスタム・ヘッダ・パラメータの追加

ObjectToJWT メソッドに渡される JOSE 配列には、JWT ヘッダに挿入するカスタム・ヘッダ・パラメータを追加できます。カスタム・ヘッダ・パラメータを追加するには、まず、それらのヘッダ・パラメータをキー/値のペアとしてダイナミック・オブジェクト内に定義します。キーはカスタム・パラメータの名前、値はそのパラメータの値です。ダイナミック・オブジェクトを定義したら、custom 添え字を使用してそのダイナミック・オブジェクトを JOSE 配列のノードに追加します。例えば、以下のコードでは、co と prod の 2 つのカスタム・パラメータが JWT ヘッダに挿入されます。

```
Set newParams={"co":"intersystems","prod":"bazbar"}
Set JOSE("custom")=newParams
// set JOSE("sigalg") and/or JOSE("encalg") and JOSE("keyalg")
Set sc=##class(%OAuth2.JWT).ObjectToJWT(.JOSE,json,%server.PrivateJWKS,%client.RemotePublicJWKS,.JWT)
```

JOSE 配列の `custom` ノードを使用して、RFC 7515 で定義されているヘッダ値をオーバーライドすることはできません。入れ子になった署名または暗号化を行う場合、カスタム・ヘッダは "内部" (署名された) トークンにのみ追加されます。

カスタム・ヘッダ・パラメータの処理

JWTToObject メソッドを使用すると、JWT を処理してヘッダを返すことができます。これらのヘッダにアクセスする方法は 2 つあります。このメソッドでは、JWT に対する署名/暗号化操作に使用するアルゴリズムが含まれる標準の JOSE ヘッダ・パラメータは、文字列の配列で返されます。また、JWT の "未加工" のヘッダはダイナミック・オブジェクトとして 6 番目のパラメータで返されます。このダイナミック・オブジェクトのキー/値ペアを解析することで、JWT ヘッダに存在するカスタムおよび標準のヘッダ・パラメータを処理できます。トークンが、入れ子になった署名および暗号化を使用して作成されている場合、このメソッドで返される未加工のヘッダは、"内部" (署名された) トークンからのものです。

代行承認

代行承認の使用法

InterSystems IRIS® データ・プラットフォームでは、ユーザ定義の承認コードの使用がサポートされています。このメカニズムを、代行承認といいます。

- 代行承認の概要
- 代行 (ユーザ定義) 承認コードの作成
- 代行承認を使用するためのインスタンスの構成
- 承認後 ー システムの状態

代行承認の概要

管理者は、代行承認により、インターシステムズのセキュリティの一部であるロール割り当ての動作に代わるカスタムのメカニズムを導入できます。例えば、ユーザ定義の承認コードが、外部データベースのユーザのロールを検索し、InterSystems IRIS にその情報を提供することができます。

代行承認を使用するには、以下の手順を実行します。

1. ZAUTHORIZE ルーチンで、代行 (ユーザ定義) 承認コードを作成します。
2. InterSystems IRIS インスタンスに対し、代行承認を使用するようインスタンスを構成します。

注釈　代行承認は、Kerberos およびオペレーティング・システム・ベースの承認でのみサポートされます。

代行認証と代行承認間の相互作用

ZAUTHORIZE.mac による代行承認は、代行認証と組み合わせて使用できません。代行認証のルーチン (″代行認証の使用法″ で説明している ZAUTHENTICATE) は、承認をサポートします。ZAUTHENTICATE を使用する場合、認証と承認のコードを分離することもできます。

重要　HealthShare® で認証を使用している場合は、インターシステムズが提供する ZAUTHENTICATE ルーチンを使用する必要があります。独自のルーチンは作成できません。

代行 (ユーザ定義) 承認コードの作成

代行承認コードの作成に関連するトピックは、以下のとおりです。

- ZAUTHORIZE.mac テンプレートからの開始
- ZAUTHORIZE シグニチャ
- ZAUTHORIZE による承認コード
- ZAUTHORIZE の返り値とエラー・メッセージ

ZAUTHORIZE.mac テンプレートからの開始

インターシステムズが提供するサンプル・ルーチン ZAUTHORIZE.mac をコピーして変更することができます。このルーチンは GitHub の Samples-Security サンプルに含まれています (https://github.com/intersystems/Samples-Security)。″InterSystems IRIS で使用するサンプルのダウンロード″ で説明されているようにサンプル全体をダウンロードすることもできますが、単に GitHub でファイルを開いて、その内容をコピーする方が簡単です。

独自の ZAUTHORIZE.mac を作成するには、以下の手順を実行します。

1. ZAUTHORIZE.mac をテンプレートとして使用するには、その内容を %SYS ネームスペースの ZAUTHORIZE.mac ルーチンにコピーして保存します。
2. ZAUTHORIZE.mac サンプル内のコメントを確認します。そこには、カスタム版のルーチンを実装する方法に関する重要なガイダンスが記載されています。
3. ユーザ・アカウントの特性を設定するには、カスタム承認コードと必要なコードを追加してルーチンを編集します。

注意　InterSystems IRIS では ZAUTHORIZE の承認コードに対する制約を行わないため、アプリケーション・プログラマは、コードが十分に安全であるかどうかを確認する必要があります。

代行承認コードのアップグレード

新しいバージョンの InterSystems IRIS にアップグレードする前に、ZAUTHORIZE.mac を確認し、現在の承認コードで新機能をサポートするための変更が必要かどうかを判断してください。

ZAUTHORIZE シグニチャ

代行承認用に構成されると、システムは承認が発生した後、自動的に ZAUTHORIZE を呼び出します。InterSystems IRIS は、必要に応じて ZAUTHORIZE シグニチャで定義されたパラメータの値を提供します。ZAUTHORIZE のシグニチャは以下のとおりです。

ObjectScript

```
ZAUTHORIZE(ServiceName, Namespace, Username, Password,
        Credentials, Properties) PUBLIC {

  // authorization code
  // optional code to specify user account properties and roles
}
```

以下はその説明です。

- ServiceName ー ユーザから InterSystems IRIS への接続を仲介しているサービスの名前を指定する文字列 (**%Service_Console** や **%Service_WebGateway** など)。
- Namespace ー 目的とする接続先の InterSystems IRIS サーバにあるネームスペースを指定する文字列。これは、スタジオや ODBC の接続など、**%Service_Bindings** のみで使用されます。その他のサービスの場合、この値は "" (引用符で囲んだ空の文字列) とする必要があります。
- Username ー 特権が決定されるユーザを指定する文字列。
- Password ー 使用しているアカウントに関連付けられたパスワードを指定する文字列。これは、Kerberos K5API 認証メカニズムのみで使用されます。その他のメカニズムの場合、この値は "" (引用符で囲んだ空の文字列) とする必要があります。
- Credentials ー 参照渡し。今回のバージョンの InterSystems IRIS では、実装されていません。
- Properties ー 参照渡し。Username で指定したアカウントの特性を指定する返り値の配列。詳細は、"ZAUTHORIZE とユーザ・プロパティ" を参照してください。

ZAUTHORIZE による承認コード

ZAUTHORIZE の目的は、認証されたユーザのロールと他の特性を確立または更新することです。承認コードのコンテンツはアプリケーションに固有です。これには、ユーザが記述した ObjectScript コード、クラス・メソッド、または $ZF コールアウトを含めることができます。

ZAUTHORIZE は、Properties 配列の値を設定することによりロール情報を指定します。この配列は、ZAUTHORIZE に参照渡しされます。通常、設定する値のソースは、ZAUTHORIZE が使用できるユーザ情報のリポジトリです。

注意　InterSystems IRIS では ZAUTHORIZE の承認コードに対する制約を行わないため、アプリケーション・プログラマは、コードが十分に安全であるかどうかを確認する必要があります。

ZAUTHORIZE とユーザ・プロパティ

Properties 配列の要素は、Username パラメータで指定されたユーザに関連付けられた属性の値を指定します。通常、ZAUTHORIZE 内のコードにより、この要素の値が設定されます。Properties 配列の要素は以下のとおりです。

- Properties("Comment") ー 任意のテキスト。
- Properties("FullName") ー ユーザの名前と姓。
- Properties("NameSpace") ー ターミナル・ログインの既定のネームスペース。
- Properties("Roles") ー ユーザが InterSystems IRIS で保持するコンマ区切りのロール・リスト。
- Properties("Routine") ー ターミナル・ログインに対して実行されるルーチン。"" の値は、ターミナルがプログラマ・モードで実行されることを示します。
- Properties("Password") ー ユーザのパスワード。
- Properties("Username") ー ユーザのユーザ名。

各要素については、この後のセクションでそれぞれ詳しく説明します。

注釈　承認後に Properties 配列のメンバの値を操作することはできません。

Comment

ZAUTHORIZE が Properties("Comment") の値を返すと、その文字列が InterSystems IRIS におけるユーザ・アカウントの Comment プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Comment の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

FullName

ZAUTHORIZE が Properties("FullName") の値を返すと、その文字列が InterSystems IRIS におけるユーザ・アカウントの Full name プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Full name の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

Namespace

ZAUTHORIZE で Properties("Namespace") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Startup Namespace プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Startup Namespace の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

InterSystems IRIS に接続すると、Startup Namespace の値 (Properties("Namespace") の値で指定されたもの) は、ローカル・アクセス (コンソール、ターミナル、Telnet など) に認証されたユーザの最初のネームスペースを決定します。Startup Namespace に値が指定されない場合、ローカル・アクセスに認証されたユーザの最初のネームスペースは以下のように決定されます。

1. USER ネームスペースが存在する場合、これが最初のネームスペースになります。
2. USER ネームスペースが存在しない場合、最初のネームスペースは %SYS ネームスペースになります。

注釈　ユーザが最初のネームスペースに対する適切な特権を保持していない場合、アクセスは拒否されます。

Password

ZAUTHORIZE で Properties("Password") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Password プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Password の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

ZAUTHORIZE がパスワードを返す場合、パスワード認証が有効であれば、システムにログインできます。これは、代行認証からパスワード認証への移行を支援するために使用可能なメカニズムですが、複数の認証メカニズムを使用する点に関して通常と同じように注意が必要です。詳細は、"カスケード認証" を参照してください。

Roles

ZAUTHORIZE で Properties("Roles") の値を設定すると、その文字列によってユーザの割り当て先の Roles が指定されます。この値はコンマ区切りのロール・リストを含む文字列です。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントに関連付けられたロールはありません。これは、管理ポータルに示されます。ユーザのロールに関する情報は、**[ユーザ編集]** ページの **[ロール]** タブおよびユーザのプロファイルで確認できます。

Properties("Roles") で返されるロールが定義されない場合、ユーザはロールに割り当てられません。

したがって、ログインしたユーザは以下のようにロールに割り当てられます。

- ロールが Properties("Roles") に示され、InterSystems IRIS インスタンスで定義される場合、ユーザはそのロールに割り当てられます。
- ロールが Properties("Roles") に示され、InterSystems IRIS インスタンスで定義されない場合、ユーザはそのロールに割り当てられません。
- ユーザは、_PUBLIC ユーザに関連付けられたロールには常に割り当てられます。また、すべてのパブリック・リソースにアクセスできます。_PUBLIC ユーザの詳細は、"_PUBLIC アカウント" を参照してください。パブリック・リソースの詳細は、"サービスとそのリソース" を参照してください。

Routine

ZAUTHORIZE で Properties("Routine") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Startup Tag^Routine プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。呼び出し元のルーチンに値が渡されない場合、ユーザ・アカウントの Startup Tag^Routine の値は NULL 文字列になり、管理ポータルの関連フィールドにはコンテンツが表示されません。

Properties("Routine") に値がある場合、この値によって、ターミナル・タイプのサービス (コンソール、ターミナル、Telnet など) でログイン後に自動的に実行するルーチンが指定されます。Properties("Routine") に値がない、または値が "" の場合、ログインは、プログラマ・モードへのアクセス権の有無に従って、プログラマ・モードでターミナル・セッションを開始します。

Username

ZAUTHORIZE で Properties("Username") の値を設定すると、その文字列が InterSystems IRIS におけるユーザ・アカウントの Name プロパティの値になります(このプロパティは、"ユーザ・アカウントのプロパティ" で説明しています)。これにより、アプリケーション・プログラマは、ログイン・プロンプトでエンドユーザが入力した内容を正規化できます (正規化されたユーザ名は大文字と小文字のみ異なります)。

Properties("Username") の値を呼び出し元のルーチンに渡す明示的な呼び出しがない場合、正規化は行われず、プロンプトでエンドユーザが入力する値が、変更されずに、そのユーザ・アカウントの Name プロパティの値としてそのまま使用されます。

ユーザ情報のリポジトリ

ZAUTHORIZE は、グローバルや外部ファイルなど、ユーザ情報のリポジトリであればどのような種類でも参照できます。認証されたユーザをこの情報で作成または更新できるように、ルーチンのコードによって Properties 配列で外部プロパティを設定します。例えば、あるリポジトリにロールやネームスペースなどの情報が含まれる一方で、ZAUTHORIZE コードは InterSystems IRIS がその情報を利用できるようにする必要があります。

リポジトリ内の情報が変更されると、このアクションを実行するコードが ZAUTHORIZE にある場合、この情報は InterSystems IRIS ユーザ情報に伝播されます。また、このようなコードがある場合、リポジトリでユーザのロールが変更されなければなりません。セッション中にユーザのロールを変更した場合、その変更は次のログインまで有効になりません。次のログインの時点で、そのユーザのロールは ZAUTHORIZE によってリセットされます。

ZAUTHORIZE の返り値とエラー・メッセージ

ルーチンは以下のいずれかの値を返します。

- 成功 ー $SYSTEM.Status.OK()。これは、ZAUTHORIZE が正常に実行されたことを示します。ルーチン内のコードにより、Username および Password に関連付けられたユーザ認証の成功、Username に関連付けられたユーザ承認の成功、またはその両方を示します。
- 失敗 ー $SYSTEM.Status.Error($$$ERRORMESSAGE)。承認が失敗したことを示します。ZAUTHORIZE がエラー・メッセージを返すと、LoginFailure イベント監査が有効であれば、このメッセージは監査ログに記録されます。エンドユーザには、$SYSTEM.Status.Error($$$AccessDenied) エラー・メッセージのみが表示されます。

ZAUTHORIZE は、システム定義またはアプリケーション固有のエラー・メッセージを返すことができます。これらのメッセージはすべて、**%SYSTEM.Status** クラスの Error メソッドを使用します。このメソッドは、$SYSTEM.Status.Error として呼び出され、エラーの状況に応じて、1 つまたは 2 つの引数を取ります。

使用可能なシステム定義のエラー・メッセージは以下のとおりです。

- $SYSTEM.Status.Error($$$AccessDenied) ー“アクセスが拒否されました”のエラー・メッセージ
- $SYSTEM.Status.Error($$$InvalidUsernameOrPassword) ー“ユーザ名またはパスワードが無効です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserNotAuthorizedOnSystem,Username) ー“ユーザ username は許可されていません”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserAccountIsDisabled,Username) ー“ユーザ username アカウントが無効です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserInvalidUsernameOrPassword,Username) ー“ユーザ username の名前またはパスワードは無効です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserLoginTimeout) ー“ログインタイムアウト”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserCTRLC) ー“ログインは中止されました”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserDoesNotExist,Username) ー“ユーザ username は存在しません”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserInvalid,Username) ー“ユーザ名 username が無効です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$PasswordChangeRequired) ー“パスワードの変更が必要です”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserAccountIsExpired,Username) ー“ユーザ username のアカウントは失効しました”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserAccountIsInactive,Username) ー“ユーザ username のアカウントはアクティブではありません”のエラー・メッセージ
- $SYSTEM.Status.Error($$$UserInvalidPassword) ー“無効なパスワードです”のエラー・メッセージ

- $SYSTEM.Status.Error($$$ServiceDisabled,ServiceName) – “サービス username のログインは無効です” のエラー・メッセージ
- $SYSTEM.Status.Error($$$ServiceLoginsDisabled) – “ログインは無効です” のエラー・メッセージ
- $SYSTEM.Status.Error($$$ServiceNotAuthorized,ServiceName) – “ユーザはサービスを許可されていません” のエラー・メッセージ

これらのエラー・コードを使用するには、ZAUTHORIZE.mac にある `#Include %occErrors` 文をアンコメントします。

カスタム・メッセージを生成するには、$SYSTEM.Status.Error() メソッドを使用して、このメソッドに `$$$GeneralError` マクロを渡し、2 番目の引数として任意のカスタム・テキストを指定します。以下に例を示します。

```
$SYSTEM.Status.Error($$$GeneralError,"Any text here")
```

エラー・メッセージが呼び出し元に返されると、そのメッセージは監査データベース (LoginFailure イベント監査が有効な場合) にログとして記録されます。表示されるエラー・メッセージは $SYSTEM.Status.Error($$$AccessDenied) のみです。ただし、`$$$PasswordChangeRequired` エラーのメッセージも表示されます。現在のパスワードから新しいパスワードへの変更をユーザに要求する場合、このエラーを返します。

代行承認を使用するためのインスタンスの構成

カスタマイズされた ZAUTHORIZE ルーチンを作成したら、次に、インスタンスの関連サービスまたはアプリケーションでそのルーチンを有効にします。手順は以下のとおりです。

1. ターミナルまたはコンソール・ウィンドウの %SYS ネームスペースから、^SECURITY ルーチンを実行します。
2. ^SECURITY で **[システムパラメータの設定]** を選択し、その下で **[認証オプション編集]** を選択し、さらにその下で **[Kerberos認証を許可]** または **[オペレーティングシステム認証を許可]** を選択します(代行承認は、これら 2 つの認証メカニズムに対してのみサポートされています)。
3. **[オペレーティングシステム認証を許可]** を選択した場合、**[O/S 認証に代行承認を許可]** を選択します。**[Kerberos認証を許可]** を選択した場合、**[Kerberos に代行承認を許可]** を選択します。

これらのいずれかを選択すると、InterSystems IRIS は %SYS ネームスペースで ZAUTHORIZE.mac ルーチンを起動します (このルーチンが存在する場合)。

重要　　InterSystems IRIS が ZAUTHORIZE を呼び出すのは、ユーザ認証後のみです。

代行承認とユーザ・タイプ

代行承認を使用する認証メカニズムでユーザが最初に InterSystems IRIS にログインすると、InterSystems IRIS は OS (オペレーティング・システム) または Kerberos のタイプのユーザ・アカウントを作成します(この値は、**[ユーザ]** ページ (**[システム管理]** > **[セキュリティ]** > **[ユーザ]**) にあるユーザのテーブルの [タイプ] 列に表示されません。)ZAUTHORIZE ルーチンは、アカウント作成の時点で、その後のログインのため、そのユーザのロールを指定します。

代行承認を使用せずにログインを試みると、ログインは失敗します。これは、代行承認のみがユーザ・タイプを OS または Kerberos として指定するためです。代行承認なしでこれらの認証メカニズムを使用する場合、ユーザはパスワード・ユーザ・タイプとして認証されます。ユーザが持つことのできるタイプは 1 つのみで、あるタイプのユーザは別のタイプに関連付けられたメカニズムを使用してログインすることができないため、ログインは失敗します(代行認証および LDAP 認証も同じ理由で失敗します)。

ユーザ・タイプの一般情報は、″ユーザ・タイプについて″ を参照してください。

承認後 – システムの状態

ユーザが正常に承認されると、InterSystems IRIS セキュリティ・データベースは次のいずれかの方法で更新されます。

1. そのユーザの最初のログインである場合、ZAUTHORIZE によって返されたプロパティを使用して、入力されたユーザ名に対するユーザ・レコードがセキュリティ・データベースに作成されます。
2. そのユーザが以前にログインしたことがある場合、この関数によって返されるプロパティを使用して、セキュリティ・データベースのユーザ・レコードが更新されます。

初めてのユーザかどうかに関係なく、ログインするプロセスは $ROLES システム変数の値を Properties("Roles") の値に設定します。ターミナル・ログインの場合、ネームスペースは Properties("NameSpace") の値に設定され、起動ルーチンは Properties("Routine") の値に設定されます。

認証に関する高度なトピック

システム変数および認証

認証の後は、以下の 2 つの変数に値が設定されます。

- $USERNAME にはユーザ名が設定されます。
- $ROLES には、このユーザが保持するロールのコンマ区切りのリストが格納されます。

$ROLES 変数を使用すると、ロールをプログラムで管理できます。

複数の認証メカニズムの使用

複数の認証メカニズムの使用が推奨される状況として、現状よりも高い厳格さを備えた認証メカニズムに移行する場合があります。例えば、認証を使用していないインスタンスで Kerberos への移行を図る場合は、以下のシナリオが考えられます。

1. 移行期間については、認証されていないアクセスと Kerberos 認証によるアクセスの両方を使用できるように、サポート対象のすべてのサービスを構成します。これにより、ユーザはどちらのメカニズムを使用しても接続できます。
2. 適切であれば、認証に Kerberos を使用するクライアント・ソフトウェアを新規にインストールします。
3. InterSystems IRIS ユーザのリストと Kerberos データベースにあるユーザのリストが同じ内容になった時点で、すべてのサービスに対し、認証されていないアクセスを無効にします。

複数の認証メカニズムを使用する場合は、通常、以下のセクションで説明するカスケード認証を組み合わせます。

カスケード認証

InterSystems IRIS では、多数の認証メカニズムをサポートしていますが、Kerberos と共に他のパスワード・ベースの認証メカニズムを使用しないことをお勧めします。また、一部の特定の状況では、インスタンスでの複数の認証メカニズムの使用が推奨されます。

サービスが複数の認証メカニズムをサポートしている場合、InterSystems IRIS では カスケード認証 によってユーザ・アクセスが管理されます。カスケード認証では、指定されているメカニズムが以下の順番で適用されてユーザの認証が行われます。

- Kerberos キャッシュ (整合性チェックや暗号化を伴う Kerberos と伴わない Kerberos のどちらも含みます)。
- OS ベース
- LDAP (LDAP 資格情報キャッシュを 2 番目にチェック)
- 代行
- インスタンス認証
- 認証なし

注釈　Kerberos プロンプトの使用が指定されているサービスで認証が失敗すると、カスケード認証は適用されません。Kerberos プロンプトと Kerberos キャッシュの両方が指定されているサービスでは、Kerberos キャッシュのみが使用されます。

例えば、以下のメカニズムによる認証をサポートしているサービスを考えます。

1. Kerberos キャッシュ
2. OS ベース
3. 認証なし

ユーザが InterSystems IRIS に接続しようとすると、そのユーザが Kerberos のチケット保証チケット (TGT) を所有するかどうかがチェックされます。このチケットがあれば、InterSystems IRIS のサービス・チケットを取得しようとする処理が実行されます。この処理が成功すると、ユーザは接続します。最初の TGT が存在しない場合、または InterSystems サービスを取得できない場合、認証が失敗し、カスケードにある次の順番のメカニズムに移ります。

InterSystems IRIS ユーザ・リストにユーザの OS ベース識別情報が含まれていれば、ユーザは接続します。ユーザの OS ベース識別情報が InterSystems IRIS ユーザ・リストにない場合は認証が失敗し、カスケードにある次の順番のメカニズムに移ります。

認証されていないアクセスが、カスケード認証にある最後の選択肢であれば、このレベルまで移ってきたユーザは全員 InterSystems IRIS にアクセスできます。

注釈　インスタンスがカスケード認証をサポートしている場合、ユーザが 2 番目またはそれ以降の認証メカニズムで認証されるということは、その認証に成功するまでにいずれかのメカニズムでログインが失敗しているということになります。%System/%Login/LoginFailure 監査イベントが有効な場合、そのようなログイン失敗の情報はインスタンスの監査ログに記録されます。

UnknownUser アカウントとの接続の確立

インスタンス認証と認証なしモードの両方が有効になっている場合、ユーザは [**ユーザ名**] プロンプトと [**パスワード**] プロンプトで Enter キーを押すだけで、認証なしモードでサービスに接続できます。この場合は、UnknownUser アカウントが使用されます。インスタンス認証のみが有効になっている場合は、[**ユーザ名**] プロンプトと [**パスワード**] プロンプトで Enter キーを押しただけでは、サービスへのアクセスが拒否されます。InterSystems IRIS では、この処理が、UnknownUser アカウントでログインしようとしたユーザが、正しくないパスワードを指定したものとして扱われるためです。

プログラムによるログイン

状況によっては、アプリケーションの実行が始まった後でユーザがログインすることが必要な場合があります。このような状況の例として、認証されていないユーザには一部の機能のみを提供し、保護された機能を提供する場合には、ユーザにログインを要求するアプリケーションがあります。

$SYSTEM.Security クラスの Login メソッドを使用すると、アプリケーションから InterSystems IRIS ログイン機能を呼び出すことができます。これには、以下の構文を使用します。

ObjectScript

```
set success = $SYSTEM.Security.Login(username,password)
```

以下はその説明です。

- success はブーリアン値で、1 は成功、0 は失敗を示します。
- username は、ログインするアカウントの名前を保持する文字列です。
- password は、username アカウントのパスワードを保持する文字列です。

有効なユーザ名とパスワードが指定され、該当のユーザ・アカウントが有効で期限切れになっていなければ、ユーザはログインできます。それに応じて $USERNAME と $ROLES が更新され、この関数からは 1 が返されます。 それ以外の場合、$USERNAME と $ROLES は変更されず、この関数からは 0 が返されます。

$SYSTEM.Security.Login. を実行した結果として、特権のチェックは行われません。その結果、プロセスがそれまで保持していた特権が失われる可能性があります。

$SYSTEM.Security.Login. には、引数が 1 つのみの以下のような形式もあります。

ObjectScript

```
set success = $SYSTEM.Security.Login(username)
```

この形式の動作は、パスワードがチェックされない点を除けば、引数が 2 つの形式とまったく同じになります。引数を 1 つ使用した形式の $SYSTEM.Security.Login が便利なのは、アプリケーションで独自の認証が実行されることから、InterSystems IRIS ユーザの識別方法をそれに応じて設定する必要がある場合です。また、あるプロセスが特定のユーザのために実行されても、そのプロセスを開始するのはそのユーザではない場合にも、この形式を使用できます。

注釈　引数を 1 つ使用した形式の Login メソッドは、制限付きのシステム機能です。

JOB コマンド、および新しいユーザ識別の確立

JOB コマンドを使用してプロセスを作成すると、その基となったプロセスからセキュリティの特性 ($USERNAME の値と $ROLES の値) が継承されます。親プロセスに保持されている、User や Added などのすべてのロールが継承されます。

しかし、新しく作成するプロセスの $USERNAME と $ROLES の値を、その親とは異なる値にすることが必要な場合があります。例えば、特定のユーザ向けに特定の時間に特定のタスクを開始する目的で、タスク・マネージャが作成されることがあります。タスク・マネージャ自体は大きな特権を持っていることが普通ですが、該当のタスクはタスク・マネージャの特権ではなく、そのタスクの対象であるユーザの特権で実行する必要があります。

以下の擬似コードは、この処理方法を示しています。

```
WHILE ConditionToTest {
    IF SomeThingToStart {
        DO Start(Routine, User)
    }
}

Start(Routine, User) {
    NEW $ROLES    // Preserve $USERNAME and $ROLES

    // Try to change username and roles
    IF $SYSTEM.Security.Login(User) {
        JOB ...
        QUIT $TEST
    }
    QUIT 0       // Login call failed
}
```

認証と管理ポータル

管理ポータルは、複数の個別の Web アプリケーションで構成されています。ポータルのメイン・ページは /csp/sys のアプリケーションと関連付けられており、他のページはさまざまな /csp/sys/* のアプリケーションと関連付けられています (/csp/sys/sec のアプリケーションと関連付けられている、セキュリティ関連のコンテンツなど)。これらのアプリケーションすべてに使用される認証メカニズムの共通セットがないと、ユーザがあるポータル・ページから別のページに移動するときに、ログイン・プロンプトが表示されたり、突然特権レベルの移行が発生したりすることがあります。

例えば、/csp/sys のアプリケーションが排他的にインスタンス認証を使用し、関連する他のポータル・アプリケーションが排他的に認証なしのアクセスを使用している場合、ユーザがあるポータル・ページから別のページに移動すると、認証なしのアクセスから認証が必要な状態に移行します。また、/csp/sys のアプリケーションはインスタンス認証のみをサポートし、他のアプリケーションは認証なしのアクセスのみをサポートしている場合、UnknownUser に特別な特権がないと、ユーザがポータルのメイン・ページからその他のページに移動するときに、何らかのアクションを実行するのに十分な特権がないという可能性もあります。

Web アプリケーションの認証メカニズムを確認および構成するには、ポータルの **[ウェブ・アプリケーション]** ページ (**[システム管理]→[セキュリティ]→[アプリケーション]→[ウェブ・アプリケーション]**) からアプリケーションを選択し、表示されたアプリケーションについて、必要に応じて、**[許可された認証方法]** で選択します (通常は、/csp/sys と /csp/sys/* で認証メカニズムの共通セットを共有するようにします)。

暗号化

インターシステムズの暗号化入門

インターシステムズの暗号化入門

InterSystems IRIS® データ・プラットフォームにはマネージド・キー暗号化のサポートが含まれています。これは保存データを保護する一連のテクノロジです。これらのテクノロジには以下のものがあります。

- ブロック・レベルのデータベース暗号化 (単にデータベース暗号化とも呼ばれます) ― すべてのデータが暗号化されるデータベースの作成と管理を可能にする管理ツールのセット。そのようなデータベースは、管理ポータルで管理されます。
- アプリケーションのデータ要素暗号化 (単にデータ要素暗号化とも呼ばれます) ― ディスクにおける格納や取得の際に個々のデータ要素 (特定のクラス・プロパティなど) の暗号化および解読を行うコードをアプリケーションで含めることができるようにする、プログラムによるインタフェース。
- 暗号化キー管理 (単にキー管理とも呼ばれます) ― データベースまたはデータ要素を暗号化するために使用するキーを作成および管理するためのツールのセット。

データベースまたはデータ要素を暗号化するためのキーは、データ暗号化キーと呼ばれ、単にキーと呼ばれる場合もあります (コンテキストが明確な場合)。各インスタンスでは、データベース暗号化用のデータ暗号化キーを最大 256 個、データ要素暗号化用のデータ暗号化キーを最大 4 個同時に有効化できます。キーを有効化することで、そのキーを暗号化操作と解読操作に利用できるようになります。

注釈　　キー・ファイル内の 1 つのキーをデータベース暗号化とデータ要素暗号化に同時に使用できます。

InterSystems IRIS では、インスタンスによるディスクに対する書き込みまたは読み取りの際に、AES (Advanced Encryption Standard) を使用して暗号化と解読が実行されます。データベースの場合、InterSystems IRIS は固定長ブロックで書き込みと読み取りを行い、単一のラベル・ブロックを除き、データベース全体が暗号化されます。この暗号化されるコンテンツには、データ自体、インデックス、ビットマップ、ポインタ、割り当てマップ、およびインクリメンタル・バックアップ・マップが含まれます。データ要素の場合、指定されたデータのみが暗号化され、暗号化キーの一意の識別子は暗号化データと共にディスクに格納されます。

暗号化と解読は最適化されていて、どのような InterSystems IRIS プラットフォームでも、暗号化と解読による影響は少なくて予測可能ですInterSystems IRIS データベース暗号化が、データベースに関連するがデータベースとは分離されている機能に与える影響の詳細は、"暗号化とデータベース関連機能" を参照してください。

キー管理タスク

キー管理タスク

キーはデータ暗号化キーの略称で、128 ビット、196 ビット、または 256 ビットのビット文字列です。これは、データの暗号化や解読を可逆的に行う暗号化アルゴリズムで使用されます。それぞれのキーには一意の識別子 (GUID とも呼ばれます) があり、InterSystems IRIS® データ・プラットフォームはこれをキー管理アクティビティの一部として表示します。

キー管理は、キーの作成、キーの有効化、キーの無効化、さまざまなアクティビティの既定のキーの割り当て、およびキーの削除に関連するアクティビティのセットです。また、キーの保存に関連する管理アクティビティも含まれます。キーは、以下の 2 つの方法のいずれかで保存できます。

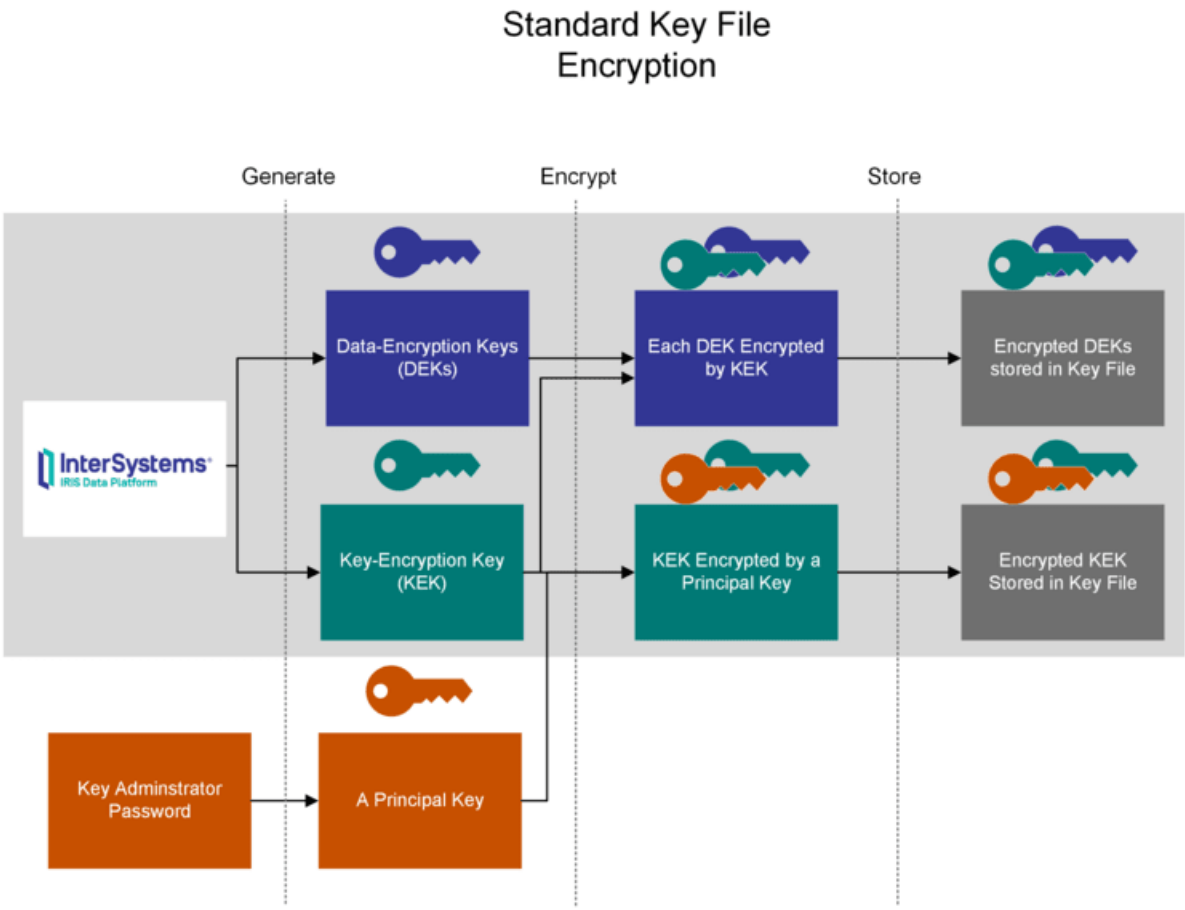
- キー・ファイル内 ― 以下のタイプのキー・ファイルがサポートされています。
  - 標準キー・ファイル ― 複数のキーを含むことができるキー・ファイル。その内容は、キー管理者のパスフレーズにより暗号化されます。これらのキー・ファイルには、最大 256 個の暗号化キーを含めることができます。詳細は、"標準キー・ファイル内のキーの管理" を参照してください。
  - KMS キー・ファイル ― AWS または Azure クラウド・サービス・プロバイダ (CSP) によって提供されるキー管理サービス (KMS) を介して暗号化されたキーを含むキー・ファイル。各 KMS キー・ファイルには、1 つのキーのみを含めることができます。前提条件と使用法については、"キー管理のための KMS の使用" を参照してください。

- KMIP サーバ ― KMIP サーバは、Key Management Interoperability Protocol (KMIP) を使用して通信を送受信できるキー管理サーバです。KMIP サーバはさまざまなサードパーティ・ベンダから発売されており、KMIP サーバの一般的な構成および使用方法は、これらのベンダによって提供されています。

注釈　ジャーナル・ファイルまたは IRISTEMP および IRISLOCALDATA データベース用の暗号化を構成する場合、これは InterSystems IRIS の起動構成の一部です。詳細は、"暗号化の起動設定の構成" を参照してください。

標準キー・ファイル内のキーの管理

キー・ファイルとは、1 つ以上のデータ暗号化キー (DEK) の暗号化コピーを保持するファイルです。キー・ファイルの管理は、キー・ファイルにおける管理者の追加や削除など、キー・ファイル自体に関連するアクティビティのセットです。特定の標準キー・ファイル内では、すべての管理者がすべてのキーにアクセスできます。すべてのキーは管理者情報と共に暗号化された形式で格納されます。各 DEK は、キー暗号化キー (KEK) を使用して個別に暗号化されます。標準キー・ファイルの管理者ごとに、一意の暗号化された KEK のコピーがあり、これは主キーを使用して暗号化されます。各主キーは個々のキー管理者のパスワードから導出されます。暗号化タスクでは有効な DEK が必要であり、InterSystems IRIS ではそのキーを解読して暗号化タスクに使用できるようにするために、管理者のユーザ名とパスワードが必要です。標準キー・ファイル・プロセスを視覚的に示した以下の図を参照してください。イタリック



標準キー・ファイルを使用した処理としては、以下のタスクがあります。

- 標準キー・ファイルの作成
- 標準キー・ファイルへのキーの追加または標準キー・ファイルからのキーの削除
- 標準キー・ファイルへの管理者の追加または標準キー・ファイルからの管理者の削除
- 標準キー・ファイルからのデータベース暗号化キーの有効化またはデータベース暗号化キーの無効化
- 標準キー・ファイルからのデータ要素暗号化キーの有効化またはデータ要素暗号化キーの無効化
- 複数インスタンス・テクノロジを使用したキーと標準キー・ファイルの管理
- インスタンスの既定の暗号化キーまたはジャーナリング暗号化キーの指定

注釈　インスタンスで起動時に複数のキーが使用される場合（例えば、ジャーナル・ファイル、監査データベース、およびその他のデータベースのキー）、これらのキーをすべて 1 つの標準キー・ファイルに配置する必要があります。これにより、これらのキーすべてがインスタンスの起動時に使用可能になります。

標準キー・ファイルの作成

標準キー・ファイルを作成すると、キーが 1 つ含まれます。標準キー・ファイルおよびその最初のキーを作成するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、**[暗号化キーファイルを作成]** ページ（**[システム管理]→[暗号化]→[新しい暗号化キーファイルを作成]**）に移動します。
2. **[暗号キー作成]** ページで、以下の値を指定します。

- **[キーファイル]** – 暗号化キーを格納する標準キー・ファイルの名前。これは、絶対パス名と相対パス名のいずれでも指定できます。

  絶対ファイル名を入力すると、標準キー・ファイルは指定したドライブの指定したディレクトリに配置されます。相対ファイル名を入力すると、標準キー・ファイルは InterSystems IRIS インスタンスの管理者ディレクトリ (InterSystems IRIS のインストール先ディレクトリの下、つまり **<install-dir>/mgr/**) に置かれます。また、このファイル名には拡張子が付加されないので、ファイル **MyKey** はそのままの名前で保存されます。このフィールドの右にある **[参照]** ボタンを使用して、標準キー・ファイルが作成されるディレクトリを選択することもできます(既存のファイル名を指定した場合、そのファイルは上書きされず、保存に失敗します)。

  警告　**<install-dir>/Mgr/Temp** にキーを格納すると、InterSystems IRIS が次回起動するときにすべて削除されます。したがって、**<install-dir>/Mgr/Temp** にはキーを絶対に格納しないでください。

- **[管理者名]** – キーを有効にできる管理者の名前。管理者を 1 人以上指定する必要があります。

  データベース暗号化機能と InterSystems IRIS のセキュリティとは互いに独立しているので、InterSystems IRIS のセキュリティで設定されていないユーザ名をここで指定してもかまいません。既定では、最初に設定した管理者名が現在のユーザ名になっています。管理者名には、Unicode 文字を使用できません。

- **[パスワード]** – このユーザのパスワード。

  データベース暗号化機能とインターシステムズのセキュリティとは互いに独立しているので、InterSystems IRIS のセキュリティでユーザが使用しているものではないパスワードをここで指定してもかまいません。このパスワードは、ディスクのどこにも格納されません。したがって、管理者は、この情報を紛失しないように注意する必要があります。

  管理者パスワードの強固さに関するガイドラインに従ったパスワードとすることをお勧めします。有効なパスワードを第三者が推測できる場合、そのパスワードのポリシーは脆弱すぎます。また、このパスワードに Unicode 文字を使用することはできません。

  重要　キー管理者のパスワードはディスクのどこにも保存されません。この情報を紛失しないよう努めることは、キー管理者の責任です。

- **[パスワード確認]** – 確認のために、このユーザのパスワードをもう一度入力します。
- **[暗号化セキュリティ・レベル]** – キーの長さ。選択肢には、128 ビット、192 ビット、および 256 ビットがあります。
- **[キーの説明]** – 最初に作成され、標準キー・ファイルに格納されているキーを説明するテキスト。このテキストは **[キーファイルに定義されている暗号化キー]** テーブルの **[説明]** 列に表示されます。

3. ページの上部の **[保存]** をクリックして、標準キー・ファイルをディスクに保存します。
4. キーを作成したら、"暗号化データのアクセスにおける偶発的な損失からの保護" の指示に従い、新たに更新された標準キー・ファイルのバックアップ・コピーを作成および保存します。

これで単一のデータベース暗号化キーおよび単一の管理者を持つ標準キー・ファイルが作成されます。ページにはキーの ID が表示されます。その文字列は、`9158980E-AE52-4E12-82FD-AA5A2909D029` などのようになります。キー ID は、InterSystems IRIS がここやその他のページに表示するキーの一意の識別子です。その場所に関係なく、キーを追跡する手段を提供します。標準キー・ファイルを保存するとどこへでも移動できる (つまり、InterSystems IRIS では標準キー・ファイルの場所によって追跡することはできない) ため、これは重要です。

キーはキー暗号化キー (KEK) を使用して暗号化されます。また、KEK の単一コピーがあり、これは管理者の主キーを使用して暗号化されます。"標準キー・ファイルへのキーの追加" の指示に従い、標準キー・ファイルにキーを追加できます。"標準キー・ファイルへの管理者の追加" の指示に従い、標準キー・ファイルに管理者を追加できます。

警告 標準キー・ファイルのバックアップ・コピーを作成および保存することを強くお勧めします。データベース暗号化キーを作成するたびに、それは再作成が不可能な一意のキーになります。同じ管理者とパスワードを使用して新しいキーを作成しても、まったく別の一意なキーが作成されます。まだ有効にしていないキーを紛失し、それをリカバリできない場合は、そのキーで保護されている暗号化データベースを読み取ることはできなくなり、そのデータは永久に失われます。

標準キー・ファイルへのキーの追加

標準キー・ファイルを使用する場合、キーを作成するには 2 つの異なる方法があります。

- 標準キー・ファイルの作成。これにより、InterSystems IRIS でキーが作成され、ファイルに配置されます。標準キー・ファイルを作成するには、"標準キー・ファイルの作成" を参照してください。
- 既存の標準キー・ファイルへのキーの追加。このセクションで説明します。

キーを既存の標準キー・ファイルに追加するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、**[暗号化キーファイル管理]** ページ (**[システム管理]**→**[暗号化]**→**[暗号化キーファイル管理]**) に移動します。
2. **[暗号化キーファイル管理]** ページの **[キーファイル]** フィールドで、キーを追加および格納する標準キー・ファイル名を入力し、[OK] をクリックします。これにより、その標準キー・ファイルの情報が表示されます。塗りつぶされた領域の下部で、**[キーファイルに定義されている暗号化キー]** テーブルに標準キー・ファイル内の 1 ～ 256 個のキーのリストが表示されます。ファイル内のキーが 255 個以下である場合、新しいキーを作成してファイルに追加できます。
3. **[キーファイルに定義されている暗号化キー]** テーブルの下にある **[追加]** ボタンをクリックして、キーを標準キー・ファイルに追加します。これにより、**[新しい暗号化キーを追加]** 画面が表示されます。
4. **[新しい暗号化キーの追加]** 画面で、以下のフィールドに値を入力します。
   - **[既存の管理者名]** – 標準キー・ファイルに関連付けられた管理者の名前(ファイルに関連付けられた管理者は、**[暗号化キーファイル管理]** ページの **[キーファイルに定義された管理者]** テーブルに表示されます)。
   - **[既存の管理者パスワード]** – この管理者のパスワード。
   - **[説明]** – キーを説明するテキスト。このテキストは **[キー・ファイルで定義されている暗号化キー]** テーブルの **[説明]** 列に表示されます。
5. [OK] をクリックして、キーを標準キー・ファイルに保存します。これにより、**[キーファイルに定義されている暗号化キー]** テーブルのキーの情報が表示されます。これにはキーの ID が含まれ、その文字列は、`9158980E-AE52-4E12-82FD-AA5A2909D029` などのようになります(キー ID は、InterSystems IRIS がここやその他のページに表示するキーの一意の識別子です。その場所に関係なく、キーを追跡する手段を提供します。標準キー・ファイルを保存するとどこへでも移動できる (つまり、InterSystems IRIS では標準キー・ファイルの場所によって追跡することはできない) ため、これは重要です)。
6. 新しいキーを標準キー・ファイルに追加したら、"暗号化データのアクセスにおける偶発的な損失からの保護" の指示に従い、新たに更新された標準キー・ファイルのバックアップ・コピーを作成および保存します。

警告 標準キー・ファイルのバックアップ・コピーを作成および保存することを強くお勧めします。データベース暗号化キーを作成するたびに、それは再作成が不可能な一意のキーになります。同じ管理者とパスワードを使用して新しいキーを作成しても、まったく別の一意なキーが作成されます。まだ有効にしていないキーを紛失し、それをリカバリできない場合は、そのキーで保護されている暗号化データベースを読み取ることはできなくなり、そのデータは永久に失われます。

標準キー・ファイルからのキーの削除

キーを標準キー・ファイルから削除するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、[**暗号化キーファイル管理**] ページ ([**システム管理**]→[**暗号化**]→[**暗号化キーファイル管理**]) に移動します。
2. [**暗号化キーファイル管理**] ページの[**キーファイル**] フィールドで、キーを削除する標準キー・ファイル名を入力し、[OK] をクリックします。これにより、その標準キー・ファイルの情報が表示されます。塗りつぶされた領域の下部で、[**キーファイルに定義されている暗号化キー**] テーブルに標準キー・ファイル内の 1 ～ 256 個のキーのリストが表示されます。ファイル内にキーが 1 つ以上ある場合、ファイルからキーを削除できます。
3. キーのテーブルで、キーの行にある [**削除**] をクリックして、そのキーを削除します。[**削除**] をクリックすると、削除の操作を確認するページが表示されます

   キーの [**削除**] ボタンが使用できない場合、これは、そのキーがファイルの既定の暗号化キーまたはジャーナル暗号化キーであるためです。キーを削除するには、まず別のキーがファイルの既定の暗号化キーまたはジャーナル暗号化キーであることを、[**既定に設定**] または [**ジャーナル設定**] をクリックして指定します。
4. 確認ダイアログで [OK] をクリックして、ファイルからキーを削除します。

警告　キーの唯一の既存コピーを削除する前に、そのコピーを使用している既存の暗号化コンテンツがないことを確実に確認してください。データを解読するために必要なキーのコピーがないと、そのキーで保護されている暗号化データは読み取り不能になり、永久に失われます。

標準キー・ファイルへの管理者の追加

管理者を既存の標準キー・ファイルに追加するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、[**暗号化キーファイル管理**] ページ ([**システム管理**]→[**暗号化**]→[**暗号化キーファイル管理**]) に移動します。
2. [**キーファイル**] フィールドで、開く標準キー・ファイルのパスとファイル名を入力し、[OK] をクリックします。[**参照**] ボタンを使用してキーを探すこともできます。ポータルで標準キー・ファイルを開くと、ファイルにリストされた管理者が記載されたテーブルが表示されます。管理者名は、定義されたときの文字に関係なく、すべて大文字で表示されます。
3. 管理者のテーブルで [**追加**] をクリックし、新しい管理者を追加します。以下のフィールドがあるページが表示されます。
   - [**既存の管理者名**] – 既にファイルにある管理者の名前。
   - [**既存の管理者パスワード**] – ファイルに既に存在する管理者に関連付けられたパスワード。
   - [**新しい管理者名**] – ファイルに追加される新しい管理者の名前。暗号化機能と InterSystems IRIS のセキュリティとは互いに独立しているので、InterSystems IRIS のセキュリティで設定されていない管理者名をここで指定してもかまいません。このユーザ名に Unicode 文字を使用することはできません。
   - [**新しい管理者パスワード**] – 新しい管理者のパスワード。管理者パスワードの強固さに関するガイドラインに従ったパスワードとすることをお勧めします。また、このパスワードは Unicode 文字を使用することはできません。データベース暗号化機能と InterSystems IRIS のセキュリティとは互いに独立しているので、InterSystems IRIS のセキュリティで設定されていないパスワードをここで指定してもかまいません。
   - [**新しい管理者パスワードの確認**] – 新しい管理者のパスワードの確認。

   これらのフィールドを入力して、[OK] をクリックします。これで新しい管理者が標準キー・ファイルに追加されました。

新しい管理者が標準キー・ファイルに追加されると、標準キー・ファイルをコピーし、各コピーを安全な場所に格納することが必要となる場合があります。さらに、キーごとに複数の管理者を作成し、そのいずれか 1 人の名前とパスワードを書面に記録して、耐火金庫などの安全な場所に保管することを強くお勧めします。ただし、ここで標準キー・ファイルのコ

ピーを作成しても、後で管理機能として新しく管理者を追加すると、新しい管理者のある標準キー・ファイルのコピーのみが最新のものになります。

注釈　新しい管理者を標準キー・ファイルに追加すると、管理者のパスワードは、ファイルに作成された管理者名のエントリに永久に関連付けられます。割り当てたパスワードは変更できません。新しいパスワードを割り当てるには、その管理者名のエントリを標準キー・ファイルから削除した後、同じ管理者名と新しいパスワードで新しいエントリを作成します。

標準キー・ファイルからの管理者の削除

管理者を標準キー・ファイルから削除するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、**[暗号化キーファイル管理]** ページ (**[システム管理]→[暗号化]→[暗号化キーファイル管理]**) に移動します。
2. **[キー・ファイル]** フィールドで、キーのパスとファイル名を入力して **[OK]** をクリックします。ファイルにリストされた管理者を含むテーブル (およびファイル内の暗号化キーのテーブル) が表示されます。
3. 管理者のテーブルで、管理者の横にある **[削除]** をクリックし、そのキーの管理者を削除します。**[削除]** をクリックすると、削除の操作を確認するページが表示されます(ファイル内に管理者が 1 人のみの場合、その管理者を削除することはできないため、**[削除]** ボタンはありません)。
4. **[OK]** をクリックして、ファイルから管理者を削除します。

標準キー・ファイルからのデータベース暗号化キーの有効化

InterSystems IRIS では、データベース暗号化で同時に 256 個までの有効なキーがサポートされています。データベース暗号化で標準キー・ファイルからキーを有効にするには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、**[データベース暗号化]** ページ (**[システム管理]→[暗号化]→[データベース暗号化]**) に移動します。有効になっているキーがある場合、ページにはそれらをリストするテーブルが表示されます。
2. このページで、**[キーを有効にする]** をクリックします。これによって、キーを有効にするためのフィールドが表示されます
3. 以下のフィールドの値を入力します。
   - **[キー・ファイル]** ー 暗号化キーを保存するファイルの名前。絶対ファイル名を入力すると、指定したドライブの指定したディレクトリにある標準キー・ファイルが検索されます。相対ファイル名を入力すると、InterSystems IRIS インスタンスの管理者ディレクトリ (InterSystems IRIS のインストール・ディレクトリの下、つまり <install-dir>/mgr/) から、標準キー・ファイルの検索が開始されます。**[参照]** ボタンを使用して標準キー・ファイルを開くダイアログを表示することもできます。
   - **[管理者名]** ー このキーが作成されたときまたは編集されたときに指定された、このキーの管理者の名前。
   - **[パスワード]** ー 指名した管理者に対して指定されているパスワード。
4. **[有効化]** ボタンをクリックします。

InterSystems IRIS により、指定したファイル内のすべてのキーの有効化が試行されます。スロットが不足していて、ファイル内のすべてのキーを有効にできない場合、できる限り多くのキーが開かれます。

キーが有効化されると、有効なキーのテーブルが **[データベース暗号化]** ページに表示されます。InterSystems IRIS により有効化されたキーそれぞれについて、有効なキーのテーブルにキーが追加され、そのキーの識別子がページで表示されます。有効なキーそれぞれに対し、以下のさまざまな操作も実行できます。

- **[既定に設定]** ー これをクリックして、新しい暗号化データベースの作成時に InterSystems IRIS でこのキーが使用されるように指定します。詳細は、"インスタンスの既定の暗号化キーまたはジャーナリング暗号化キーの指定" を参照してください。

- **[ジャーナルの設定]** – これをクリックして、InterSystems IRIS でそのキーを使用してジャーナル・ファイルを暗号化するように指定します。詳細は、"インスタンスの既定の暗号化キーまたはジャーナリング暗号化キーの指定" を参照してください。
- **[無効]** – これをクリックして、このキーを無効にします。詳細は、"データベース暗号化キーの無効化" を参照してください。

注釈　キーのテーブルにファイルやパスの情報は表示されません。標準キー・ファイルが作成されると、十分な特権のあるオペレーティング・システム・ユーザは標準キー・ファイルを移動できるので、InterSystems IRIS ではオペレーティング・システムの場所に関して正確な情報がなくなる場合があり、信頼できるのは、メモリにある有効なキーの GUID の正確性のみとなるためです。2 番目以降のキーを有効化する場合は、まず、現在有効化されているキーの識別子をメモしてください。これによって新しいキーを識別できます。

標準キー・ファイルからのデータ要素暗号化キーの有効化

InterSystems IRIS では、データ要素暗号化で一度に 4 つまでの有効なキーがサポートされています。データ要素暗号化でキーを有効にするには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、**[データ要素暗号化]** ページ (**[システム管理]** > **[暗号化]** > **[データ要素暗号化]**) に移動します。有効になっているキーがある場合、ページにはそれらをリストするテーブルが表示されます。
2. **[データ要素暗号化]** ページで、**[キーを有効にする]** を選択します。これによって、キーを有効にするためのフィールドが表示されます。キー有効化が利用できない場合、データ要素暗号化キーは既に 4 つ有効になっています。
3. 以下のフィールドの値を入力します。
   - **[キー・ファイル]** – 暗号化キーを保存するファイルの名前。絶対ファイル名を入力すると、指定したドライブの指定したディレクトリにある標準キー・ファイルが検索されます。相対ファイル名を入力すると、InterSystems IRIS インスタンスの管理者ディレクトリ (InterSystems IRIS のインストール・ディレクトリの下、つまり <install-dir>/mgr/) から、標準キー・ファイルの検索が開始されます。
   - **[管理者名]** – このキーが作成されたときまたは編集されたときに指定された、このキーの管理者の名前。
   - **[パスワード]** – 指名した管理者に対して指定されているパスワード。
4. **[有効化]** ボタンをクリックします。

InterSystems IRIS により、指定したファイル内のすべてのキーの有効化が試行されます。スロットが不足していて、ファイル内のすべてのキーを有効にできない場合、できる限り多くのキーが開かれます。

キーが有効化されると、有効なキーのテーブルが **[データ要素暗号化]** ページに表示されます。InterSystems IRIS により有効化されたキーそれぞれについて、有効なキーのテーブルにキーが追加され、そのキーの識別子がページで表示されます。

注釈　キーのテーブルにファイルやパスの情報は表示されません。標準キー・ファイルが有効になると、十分な特権のあるオペレーティング・システム・ユーザはキーを移動できるので、InterSystems IRIS ではオペレーティング・システムの場所に関して正確な情報がなくなる場合があり、信頼できるのは、メモリにある有効なキーの GUID の正確性のみとなるためです。2 番目以降のキーを有効化する場合は、まず、現在有効化されているキーの識別子をメモしてください。これによって新しいキーを識別できます。

複数インスタンス・テクノロジを使用したキーと標準キー・ファイルの管理

InterSystems IRIS のクラスタ内で暗号化データベースやジャーナル・ファイルを使用している場合、クラスタのすべてのノード上の InterSystems IRIS インスタンスは、単一のデータベース暗号化キーを共有する必要があります。

InterSystems IRIS ミラーに属しているインスタンスに対してジャーナル・ファイルの暗号化を有効にする前に、"ミラー内ジャーナル暗号の有効化" を参照して重要な情報を確認してください (データベース暗号化についてはミラーリングに関する要件はありません)。

単一のキーの共有を有効にする方法は 2 つあります。

- すべてのインスタンスは、1 つのクラスタ・ノードまたはミラー・メンバにある単一の標準キー・ファイルを共有しています。

  この場合、標準キー・ファイルの単一のコピーを変更すれば、すべてのノードまたはメンバでその変更を認識できます。ただし、標準キー・ファイルを保持しているホストが他のノードまたはメンバで利用できなくなると、標準キー・ファイルからキーを読み取れず、InterSystems IRIS インスタンスを正しく再起動できなくなる可能性があります。

- 標準キー・ファイルのコピーをクラスタ・ノードごとまたはミラー・メンバごとに保持します。

  この場合、標準キー・ファイルを変更したら、同じキーを含む標準キー・ファイルのコピーを他のすべてのノードまたはメンバに配布します。この方法では、(通常は負荷が小さい) 標準キー・ファイルを管理する負担が大きくなりますが、各 InterSystems IRIS インスタンスで起動時にキーを確実に利用できます。

重要　標準キー・ファイルが単一または複数のどちらの場合でも、データベース暗号化キー自体はすべてのインスタンスに共通です。

単一の標準キー・ファイルの使用

単一の標準キー・ファイルを使用するには、以下の手順に従います。

1. データベース暗号化キーを 1 台のノードまたはメンバ上に作成します。この手順の詳細は、"標準キー・ファイルの作成" を参照してください。
2. "暗号化データのアクセスにおける偶発的な損失からの保護" の指示に従い、このキーを保護します。

   注意　これらの予防措置を怠ると、暗号化データベースまたはジャーナル・ファイルを読み取れず、そのデータが永久に失われる結果になることがあります。

3. 無人で起動するように各 InterSystems IRIS インスタンスを構成して、InterSystems IRIS に標準キー・ファイルのパスを提供します。この手順の詳細は、"無人のキー有効化による起動" を参照してください。

すべての InterSystems IRIS インスタンスは同じキーを使用するため、暗号化されたデータをインスタンス間で読み取れます。標準キー・ファイルを変更すると、すべてのインスタンスでその変更を認識できます。

複数の標準キー・ファイルの使用

標準キー・ファイルのコピーを複数使用するには、以下の手順に従います。

1. データベース暗号化キーを 1 台のノードまたはメンバ上に作成します。この手順の詳細は、"標準キー・ファイルの作成" を参照してください。
2. "暗号化データのアクセスにおける偶発的な損失からの保護" の指示に従い、このキーを保護します。

   注意　これらの予防措置を怠ると、暗号化データベースまたはジャーナル・ファイルを読み取れず、そのデータが永久に失われる結果になることがあります。

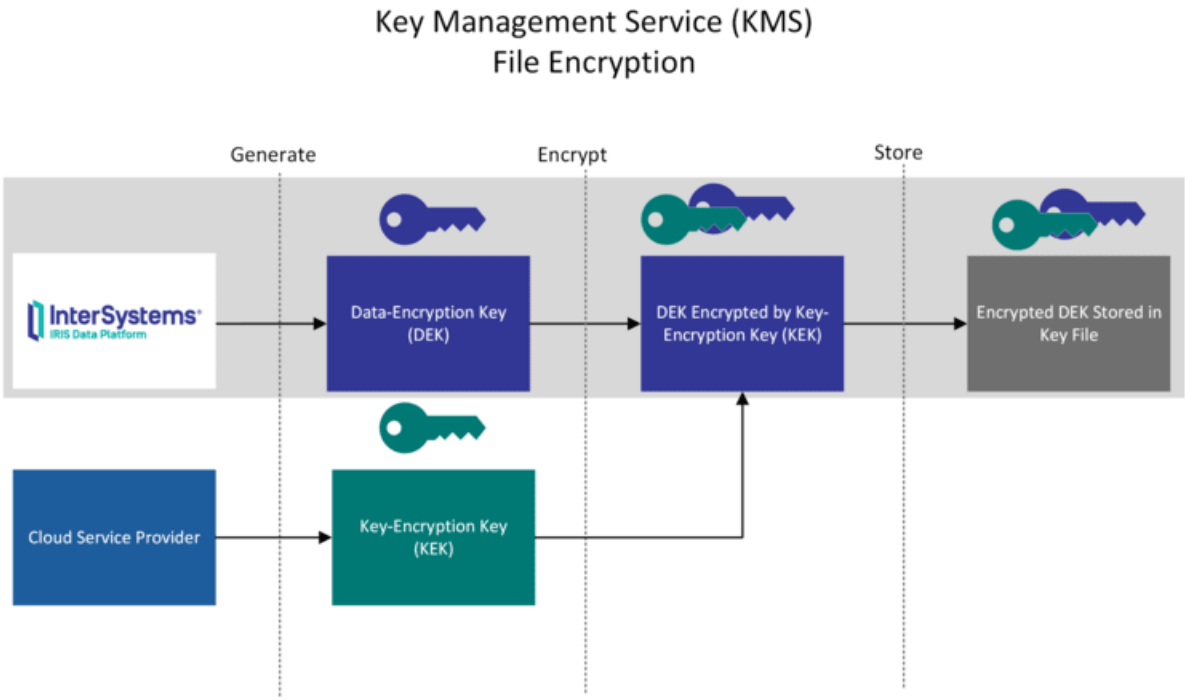
3. 標準キー・ファイルのコピーを残りのノードまたはメンバごとに作成します。
4. ノードまたはメンバごとに次の手順を実行します。
   a. 標準キー・ファイルのコピーを取得して、マシン上の安全で安定した場所にそのコピーを置きます。
   b. 無人で起動するように各 InterSystems IRIS インスタンスを構成します。この手順の詳細は、"無人のキー有効化による起動" を参照してください。

標準キー・ファイルの各コピーには同じキーが含まれるため、すべての InterSystems IRIS インスタンス間で暗号化されたデータを読み取ることができます。各 InterSystems IRIS インスタンスのマシン上に標準キー・ファイルがあるので、InterSystems IRIS の再起動時に確実に標準キー・ファイルを使用できます。管理者の追加や削除などで標準キー・ファ

イルを変更した場合は、各マシンにその標準キー・ファイルの新しいコピーを配布して、(古いコピーと同じ場所に新しいコピーを置いた場合でも) 新しいコピーを使用して無人で起動するように各 InterSystems IRIS インスタンスを再構成する必要があります。

キー管理のための KMS の使用

InterSystems IRIS により、クラウド・サービス・プロバイダ (CSP) によって提供されるキー管理サービス (KMS) を使用できます。KMS キーは、標準キー・ファイルの暗号化プロセスと同じように、キー暗号化キー (KEK) として機能します。つまり、KEK はデータ暗号化キー (DEK) を暗号化し、その後この DEK は KMS キー・ファイルに格納されます。このプロセスの概要を以下の図に示します。



前提条件

- KMS プロバイダに固有のコマンド行インタフェース (CLI) をインストールします。詳細は、CSP によって提供される KMS ドキュメントを参照してください。
- CSP に対して認証を受ける際に InterSystems IRIS インスタンスを表すユーザ/サービス・プリンシパルを作成および構成します。このユーザには、キーに関連付けられたキー・ポリシーを通じて適切なキー・アクセス・ポリシーが構成されている必要があります。Azure の場合、これはキーに関連付けられたキー・ボールトのアクセス構成を通じて、ボールトのアクセス・ポリシーまたは RBAC 許可モデルを使用して行われます。構成した InterSystems IRIS のユーザ・プリンシパルまたはサービス・プリンシパルを使用し、CLI を通じて CSP に対して認証できることを確認します。詳細は、CSP によって提供される KMS ドキュメントを参照してください。
- CSP サーバで KMS キーを作成します。AWS の場合は、対称鍵を作成します。詳細は、AWS のキーの作成に関するページを参照してください。Azure の場合は、RSA キーを作成します。詳細は、Azure Key Vault のキーについてのページを参照してください。このキーのキー ID は KMS キー・ファイルの作成に必要であるため、使用できるようにしておいてください。

KMS とのやり取り

暗号化に KMS を使用してキー管理タスクを実行するには、`^EncryptionKey` ルーチンまたは InterSystems IRIS KMS API のいずれかを呼び出すことができます。これらはどちらも InterSystems IRIS ターミナルの `%SYS` ネームスペースにあります。`^EncryptionKey` ルーチンの場合、以下のコマンド行を入力します。

```
%SYS>do ^EncryptionKey
```

^EncryptionKey には、以下の KMS キー管理タスクを実行できるようにするメニュー駆動型のインタフェースが含まれます。

- キーの作成。
- データベース暗号化キーの有効化。
- データ要素暗号化キーの有効化。
- KMS キー・ファイルを使用した無人キー有効化の構成。

^EncryptionKey の詳細は、"コマンド行セキュリティ管理ユーティリティ" を参照してください。

^EncryptionKey ルーチンを使用した新しい KMS 暗号化キーの作成

^EncryptionKey ルーチンを使用して新しい KMS 暗号化キーを作成するには、以下の手順を実行します。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに切り替えます。

   ```
   >zn "%SYS"
   ```

3. ^EncryptionKey を実行します。

   ```
   >do ^EncryptionKey
   ```

4. ^EncryptionKey で、オプション 1 [**新しい暗号化ファイルの作成**] を選択し、プロンプトでキー・ファイルの名前を入力します。拡張子は付加されないため、入力した名前がそのままファイル名になります。
5. KMS を使用するには、[Use KMS?] プロンプトで **y** と入力します。
6. 必要に応じて説明を入力します。
7. キー・サイズを選択します。
8. オプション：標準キー・ファイルにバックアップする場合は、以下の手順を実行します。
   a. プロンプトが表示されたら、バックアップの標準キー・ファイルの名前とキーの説明を入力します。
   b. キー管理者のユーザ名とパスワードを入力します。

      データベース暗号化機能とインターシステムズのセキュリティとは互いに独立しているので、InterSystems IRIS のセキュリティでユーザが使用しているものではないパスワードをここで指定してもかまいません。このパスワードは、ディスクのどこにも格納されません。したがって、管理者は、この情報を紛失しないように注意する必要があります。

      管理者パスワードの強固さに関するガイドラインに従ったパスワードとすることをお勧めします。有効なパスワードを第三者が推測できる場合、そのパスワードのポリシーは脆弱すぎます。また、このパスワードに Unicode 文字を使用することはできません。
9. 目的の KMS (AWS または Azure) を選択します。
10. [**サーバ・キー ID**] プロンプトで、使用する KMS キーのキー ID を入力します。これは、以前 KMS で構成したキーの ID です。
11. Azure を使用している場合は、この手順をスキップしてください。AWS を使用している場合は、以下の手順に従います。
    a. リージョンを入力します (us-east-2 など)。

b. オプション：[Environment variable key] プロンプトで AWS 環境変数を入力できます。目的の変数ごとに、[Environment variable key] プロンプトで変数キーを入力します。次に、[Environment variable value] プロンプトで値を入力します。AWS 環境変数の詳細は、AWS のドキュメントを参照してください。環境変数を使用する 1 つの理由は、コマンド行を介した CSP に対する認証で、共有認証情報ファイル (AWS_SHARED_CREDENTIALS_FILE) を使用するためです。

12. 新しい KMS キー・ファイルとデータ暗号化キーが作成されます。ディスプレイには以下の情報がエコー・バックされます。

暗号化キーファイルが作成されました

データ暗号化キーを含む暗号化ファイルの名前。

Encryption key backup file created

バックアップ標準キー・ファイルの名前 (指定されている場合)。

Encryption key created via KMS

InterSystems IRIS 内のデータの暗号化に使用されるキー。

13. オプション：既定で新しい KMS キー・ファイルを使用するように、InterSystems IRIS 起動オプションを構成します。

a. 4 [InterSystems IRIS 開始オプションを構成] を選択します。

b. 3 [キーファイルを使用した無人キー有効化] を選択し、KMS キー・ファイル名を入力します。AWS を使用していない場合は、[Environment variable key] プロンプトを無視してください。

無人での起動の詳細は、"無人のキー有効化による起動" を参照してください。

InterSystems IRIS KMS API を使用した新しい KMS 暗号化キーの作成

API を使用して新しい KMS 暗号化キー・ファイルを作成するには、$System.Encryption.KMSCreateEncryptionKey() メソッドを使用します。これには ^EncryptionKey ルーチンと同じ KMS 前提条件があります。以下の使用方法に従います。

```
%SYS>set KeyID =
$System.Encryption.KMSCreateEncryptionKey(File,Server,ServerKeyID,KeyLength,.Backup,Region,Description,.Env,.Status)
```

KeyID を取得して格納します。これは、目的のデータ暗号化キーの特定およびインポートに必要なためです。これは、保存データの暗号化とデータ要素の暗号化に機能します。

このメソッド引数は以下のように定義されます。

- **File** – 。作成するキー・ファイルの名前。
- **Server** – 。KMS CSP サーバの名前。現在受け入れられる値は、"AWS" と "Azure" (大文字と小文字は区別されません) です。
- **ServerKeyID** – 。サーバ上の主キーのキー ID。
- **KeyLength** – 。データ暗号化キーとキー暗号化キーのバイト単位の長さ。16、24、または 32 を指定してください。
- **Backup** – 。バックアップ・キー・ファイル作成のための情報。指定する場合は、以下のエントリを含める必要があります。
  - Backup("File") – 。バックアップ・キー・ファイルの名前。
  - Backup("Username") – 。バックアップ・キー・ファイルの最初の暗号化キー管理者の名前。
  - Backup("Password") – 。バックアップ・キー・ファイルの最初の暗号化キー管理者のパスワード。

注釈　この値は必ずユーザ・プロンプトから取得する必要があります。アプリケーション・コードに埋め込まないでください。

- `Backup("Desc")` ― 。キーの説明。

- **`Region`** ― `AWS` 。リージョンの名前 (`us-east-2` など)。
- **`Description`** ― 。キーの説明。
- **`Env`** ― `AWS` 。環境変数情報(`Env("AWS_CONFIG_FILE")`、`Env("AWS_SHARED_CREDENTIALS_FILE")` など)。
- **`Status`** ― 。メソッドのステータスを返します。

成功した場合、このメソッドは新しい暗号化キーの一意の識別子を返します。AWS と Azure での使用方法は、以下の例を参照してください。API では、呼び出し元ユーザに、**`%Admin_Secure:U`** 特権と **`%SYS`** ネームスペースへのアクセス権が必要です。

AWS KMS API 呼び出しの場合：

```
%SYS>w ^KeyIDaws
604cae51-139d-4b88-b8ac-8b303446ebe7
%SYS>s
KeyID=$System.Encryption.KMSCreateEncryptionKey("KMSTestAWS.key","AWS",^KeyIDaws,16,,"us-east-2","aws
 test key",,.rc)
%SYS>w rc," ",KeyID
1 3C978FFD-DEA8-4393-A454-BC06B311D545
```

Azure KMS API 呼び出しの場合：

```
%SYS>w ^KeyIDaz
https://test.vault.azure.net/keys/testkey/3ab1ba844c0b407c9b6063cefe5053dd
%SYS>s KeyID=$System.Encryption.KMSCreateEncryptionKey("KMSTestAZ.key","azure",^KeyIDaz,16,,,,,.rc)
%SYS>w rc," ",KeyID
1 8DC5555E-A464-4A59-9B3A-FD06857E5056
```

Key Management Interoperability Protocol (KMIP) を使用したキーの管理

インターシステムズは、KMIP サーバを使用したデータベース暗号化キー管理をサポートしています。KMIP の使用には以下のタスクが含まれます。

- KMIP サーバ構成の作成、編集、または削除
- KMIP サーバ構成のリスト
- KMIP サーバ構成の詳細のリスト
- KMIP サーバ上のキーの作成またはKMIP サーバ上のキーの削除
- KMIP サーバ上のキーのリスト
- KMIP サーバからのデータベース暗号化キーの有効化またはデータベース暗号化キーの無効化
- KMIP サーバからのデータ要素暗号化キーの有効化またはデータ要素暗号化キーの無効化
- KMIP サーバからキー・ファイルへのキーのコピー
- インスタンスの既定の暗号化キーまたはジャーナリング暗号化キーの指定

注釈
- InterSystems IRIS は、KMIP プロトコルのバージョン 1.0–2.1 をサポートしています。
- KMIP のアクティビティは、InterSystems IRIS の macOS インスタンスではサポートされません。

KMIP サーバ構成の作成

InterSystems IRIS と KMIP サーバとの間の接続を確立する場合、KMIP サーバのプロパティを定義してそれを InterSystems IRIS インスタンス内で表す KMIP サーバ構成を作成します。KMIP サーバ構成を作成するには、以下の手順に従います。

1. ベンダの指示に従って KMIP サーバを設定します。

   注意　KMIP サーバを構成する場合、ベンダの指示に従って適切なバックアップ手順をすべて実行してください。キーのバックアップ・コピーがないと、データが永久に失われることがあります。

   サーバを設定したら、InterSystems IRIS で KMIP サーバ構成を設定できます。

2. KMIP サーバ構成を設定するには、以下が必要です。
   - KMIP サーバに対する認証局 (CA) の証明書。これは、信頼された CA である必要があります。KMIP サーバを提供するベンダからこの証明書を受け取るか、そのベンダの指示に従って証明書を入手する必要があります。
   - KMIP サーバと通信する InterSystems IRIS の各インスタンスの公開鍵証明書と秘密鍵。証明書は、信頼された CA によって発行されている必要があります。KMIP サーバを提供するベンダからこの証明書と秘密鍵を受け取るか、そのベンダの指示に従ってそれらを入手する必要があります。
   - KMIP サーバに関する以下の情報
     - その完全修飾 DNS 名または IP アドレス
     - 接続を受け付けるポート番号
     - サーバがサポートする KMIP プロトコルのバージョン
     - クライアントのために必要な TLS 設定

3. KMIP サーバと通信する InterSystems IRIS インスタンス上で、KMIP サーバに対してそのインスタンスを表す TLS 構成を作成します。
   a. ポータルで、**[SSL/TLS 構成]** ページ (**[ホーム]**→**[システム管理]**→**[セキュリティ]**→**[SSL/TLS 構成]**) に移動します。
   b. **[SSL/TLS 構成]** ページで **[新規構成の作成]** ボタンをクリックして、**[新規 SSL/TLS 構成]** ページを表示します。
   c. **[新規 SSL/TLS 構成]** ページで、TLS 構成を設定します。以下に示すフィールドで、以下のように値を指定または選択します。
      - **[有効]** ― このチェック・ボックスにチェックを付けます。
      - **[タイプ]** ― **[クライアント]** を選択します。

      他のフィールド (**[サーバ証明書の検証]**、**[このクライアントの認証情報]**、および **[暗号方式設定]** の各フィールド) の値は、KMIP サーバの要件によって異なります。**[このクライアントの認証情報]** フィールドの値は、クライアントの証明書、クライアントの秘密鍵、および KMIP サーバを提供するベンダから受け取った CA 証明書によって異なります。

      ここで TLS 構成を作成する方法の詳細は、"TLS 構成の作成または編集" を参照してください。

4. KMIP サーバに対して構成を作成します。
   a. ターミナルを起動し、十分な特権を持つユーザとしてログインします。
   b. ターミナル・プロンプトで、**%SYS** ネームスペースに移動します。

```
>set $namespace="%SYS"
```

c. ^SECURITY を実行します。

```
%SYS>do ^SECURITY
```

d. ^SECURITY で、オプション [14]、[KMIP サーバの設定] を選択します。

e. [KMIP サーバの設定] の選択肢で、オプション [1]、[KMIP サーバの作成] を選択します。

f. [KMIP サーバの作成] プロンプトで、以下の値を指定します。

- [作成する KMIP サーバ?]— KMIP サーバ構成の名前。
- [説明?]— 説明テキスト。
- [サーバ・ホストの DNS 名?]— KMIP サーバの完全修飾 DNS 名または IP アドレス。
- [TCP ポート番号?]— KMIP サーバが接続を受け付けるポート番号。
- [OASIS KMIP プロトコル・バージョン?]— KMIP サーバでサポートされているプロトコル・バージョンに関連する数字。これは、KMIP サーバを提供するベンダから受け取った情報の一部です。
- [SSL/TLS 構成名?]— 前の手順で作成した TLS 構成の名前。

  注釈　ここに入力する値の大文字と小文字は、定義されている TLS 構成名と一致する必要があります。

- [非ブロッキング I/O?]— KMIP サーバとの接続で非ブロッキング I/O を有効にするかどうか。[はい] を選択して、非ブロッキング I/O を有効にすることをお勧めします。

  非ブロッキング I/O が有効の場合、[I/O タイムアウト (秒)?] プロンプト (以下を参照) で指定したタイムアウトの経過後に、制御がアプリケーションに戻ります。非ブロッキング I/O が無効の場合は、オペレーティング・システムのタイムアウト (これは発生しない可能性があります) の経過後に制御がアプリケーションに戻ります。
- [自動再接続?]— 接続が解除された場合に InterSystems IRIS が KMIP サーバに再接続するかどうか。インターシステムズでは、[いいえ] を選択して、接続が解除された場合に自動的に再接続を試行しないようにすることをお勧めします。
- [I/O タイムアウト (秒)?]— KMIP サーバとの接続がタイムアウトするまでの時間 (秒)。これは、構成で非ブロッキング I/O が有効になっている場合にのみ関係があります。
- [KMIP メッセージのログ?]— KMIP サーバに送信するメッセージを InterSystems IRIS がログに記録するかどうか。メッセージをログに記録する場合、ログは <install-dir>/mgr/kmipcmd.log ファイルに保存されます。
- [SSL/TLS のデバッグ?]— InterSystems IRIS が TLS デバッグ情報をログに記録するかどうか。情報をログに記録する場合、情報は <install-dir>/mgr/kmipssl.log ファイルに保存されます。

g. KMIP サーバのプロパティの入力を求めるプロンプトの後、[KMIP サーバの作成の確認] プロンプトで、KMIP サーバを作成することを確認します。

注釈　複数の KMIP サーバを使用したり、複数の構成を持つ単一の KMIP サーバを使用したりできます。最後に有効化した構成が既定になります。

KMIP サーバ構成の編集

既存の KMIP サーバ構成のプロパティの値を変更するには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

```
>set $namespace="%SYS"
```

3. ^SECURITY を実行します。

```
%SYS>do ^SECURITY
```

4. ^SECURITY で、オプション [14]、[KMIP サーバの設定] を選択します。
5. [KMIP サーバの設定] の選択肢で、オプション [2]、[KMIP サーバの編集] を選択します。
6. [KMIP サーバの編集] プロンプトで、編集する構成の名前を入力します。
7. ^SECURITY により、KMIP サーバ構成の作成時と同じプロパティの入力を求めるプロンプトが表示されます。構成のプロパティの既存値が既定値として使用されます。必要に応じて、これらの値を変更します。
8. KMIP サーバのプロパティの入力を求めるプロンプトの後、[KMIP サーバ <servername> の変更の確認] プロンプトで、KMIP サーバのプロパティに加えた編集を確認します。

KMIP サーバ構成の削除

KMIP サーバ構成を削除するには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

```
>set $namespace="%SYS"
```

3. ^SECURITY を実行します。

```
%SYS>do ^SECURITY
```

4. ^SECURITY で、オプション [14]、[KMIP サーバの設定] を選択します。
5. [KMIP サーバの設定] の選択肢で、オプション [5]、[KMIP サーバの削除] を選択します。
6. [削除する KMIP サーバ?] プロンプトで、削除する構成の名前を入力します。
7. プロンプトが表示されたら、削除を確認します。

KMIP サーバ構成のリスト

InterSystems IRIS インスタンスの KMIP サーバ構成をリストするには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

```
>set $namespace="%SYS"
```

3. ^SECURITY を実行します。

```
%SYS>do ^SECURITY
```

4. ^SECURITY で、オプション [14]、[KMIP サーバの設定] を選択します。
5. [KMIP サーバの設定] の選択肢で、オプション [3]、[KMIP サーバのリスト] を選択します。

^SECURITY により、KMIP サーバの既存の構成のリストが、現在使用されているかどうかに関係なく、名前別に表示されます。

KMIP サーバ構成の詳細のリスト

特定の KMIP サーバ構成の詳細を表示するには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

```
>set $namespace="%SYS"
```

3. ^SECURITY を実行します。

```
%SYS>do ^SECURITY
```

4. ^SECURITY で、オプション [14]、[KMIP サーバの設定] を選択します。
5. [KMIP サーバの設定] の選択肢で、オプション [4]、[KMIP サーバの詳細リスト] を選択します。
6. [どの KMIP 構成を表示しますか?] プロンプトで、KMIP サーバ構成の名前を入力します。

^SECURITY により、指定した構成のプロパティと各プロパティの値のリストが表示されます。

KMIP サーバ上のキーの作成

KMIP サーバ上にデータ暗号化キーを作成するには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

```
>set $namespace="%SYS"
```

3. ^EncryptionKey を実行します。

```
%SYS>do ^EncryptionKey
```

4. ^EncryptionKey で、オプション [5]、[KMIP サーバの管理] を選択します。
5. プロンプトが表示されたら、キーを作成する KMIP サーバの構成の名前を入力します。
6. 実行するアクションを選択する次のプロンプトで、オプション [2]、[KMIP サーバ上に新規キーを作成] を選択します。
7. 次のプロンプトで、キー長を選択します。

^EncryptionKey ルーチンによってキーが作成され、そのキー ID が表示されます。新しく作成されたキーは既定では有効化されていません。キーを有効化するには、"KMIP サーバからのデータベース暗号化キーの有効化" を参照してください。

重要　　インターシステムズでは、後で参照できるようにキー ID を記録しておくことをお勧めします。

KMIP サーバ上のキーの削除

KMIP サーバ上の暗号化キーを削除するには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

```
>set $namespace="%SYS"
```

3. ^EncryptionKey を実行します。

```
%SYS>do ^EncryptionKey
```

4. ^EncryptionKey で、オプション [5]、[KMIP サーバの管理] を選択します。
5. プロンプトが表示されたら、キーを削除する KMIP サーバの構成の名前を入力します。

6. 実行するアクションを選択する次のプロンプトで、オプション [3]、[KMIP サーバ上の既存キーの破棄] を選択します。
7. KMIP サーバ上のキーがリストされ、削除するキーを指定するよう求められます。[キーの選択] プロンプトでキーを指定します。

   警告　キーの唯一の既存コピーを削除する前に、そのコピーを使用している既存の暗号化コンテンツがないことを確実に確認してください。データを解読するために必要なキーのコピーがないと、そのキーで保護されている暗号化データは読み取り不能になり、永久に失われます。

8. プロンプトが表示されたら、キーを削除することを確認します。

KMIP サーバからキーが削除されます。

KMIP サーバ上のキーのリスト

KMIP サーバ上の暗号化キーをリストするには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

   ```
   >set $namespace="%SYS"
   ```

3. ^EncryptionKey を実行します。

   ```
   %SYS>do ^EncryptionKey
   ```

4. ^EncryptionKey で、オプション [5]、[KMIP サーバの管理] を選択します。
5. プロンプトが表示されたら、キーをリストする KMIP サーバの構成の名前を入力します。
6. 次のプロンプトで、オプション [1]、[KMIP サーバ上のキーのリスト] を選択します。

KMIP サーバ上にあるすべてのキーのリストが表示されます。

KMIP サーバからのデータベース暗号化キーの有効化

KMIP サーバからデータベース暗号化キーを有効化するには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

   ```
   >set $namespace="%SYS"
   ```

3. ^EncryptionKey を実行します。

   ```
   %SYS>do ^EncryptionKey
   ```

4. ^EncryptionKey で、オプション [3]、[データベース暗号化] を選択します。
5. [データベース暗号化] の選択肢で、オプション [1]、[データベース暗号化キーの有効化] を選択します。
6. [データベース暗号化キーの有効化] の選択肢で、オプション [2]、[KMIP サーバの使用] を選択します。

   注釈　このプロンプトが表示されない場合、インスタンスに KMIP サーバ構成がありません。このプロセスの手順は、"KMIP サーバ構成の作成" を参照してください。

7. プロンプトが表示されたら、キーを有効化する KMIP サーバの構成の名前を入力します。
8. KMIP サーバ上のキーがリストされ、有効化するキーを指定するよう求められます。[キーの選択] プロンプトでキーを指定します。

キーが有効化され、その ID が表示されます。

InterSystems IRIS により有効化されたキーそれぞれについて、有効なキーのテーブルにキーが追加され、そのキーの識別子が [**データベース暗号化**] ページ ([**システム管理**] > [**暗号化**] > [**データベース暗号化**]) に表示されます。

注釈　キーのテーブルにファイルやパスの情報は表示されません。2 番目以降のキーを有効化する場合は、まず、現在有効化されているキーの識別子をメモしてください。これによって新しいキーを識別できます。

KMIP サーバからのデータ要素暗号化キーの有効化

InterSystems IRIS では、データ要素暗号化で一度に 4 つまでの有効なキーがサポートされています。KMIP サーバからデータ要素暗号化キーを有効にするには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、**%SYS** ネームスペースに移動します。

   ```
   >set $namespace="%SYS"
   ```

3. ^EncryptionKey を実行します。

   ```
   %SYS>do ^EncryptionKey
   ```

4. ^EncryptionKey で、オプション [**4**]、[**アプリケーションのデータ要素暗号化**] を選択します。
5. [**アプリケーションのデータ要素暗号化**] の選択肢で、オプション [**1**]、[**データ要素暗号化キーの有効化**] を選択します。
6. [**データ要素暗号化キーの有効化**] の選択肢で、オプション [**2**]、[**KMIP サーバの使用**] を選択します。

   注釈　このプロンプトが表示されない場合、インスタンスに KMIP サーバ構成がありません。このプロセスの手順は、"KMIP サーバ構成の作成" を参照してください。

7. KMIP サーバのプロンプトで、キーを有効化する KMIP サーバの構成の名前を入力します。
8. KMIP サーバ上のキーがリストされ、有効化するキーを指定するよう求められます。[**キーの選択**] プロンプトでキーを指定します。

キーが有効化され、その ID が表示されます。

InterSystems IRIS により有効化されたキーそれぞれについて、有効なキーのテーブルにキーが追加され、そのキーの識別子が [**データ要素暗号化**] ページ ([**システム管理**] > [**暗号化**] > [**データ要素暗号化**]) に表示されます。

注釈　キーのテーブルにファイルやパスの情報は表示されません。2 番目以降のキーを有効化する場合は、まず、現在有効化されているキーの識別子をメモしてください。これによって新しいキーを識別できます。

KMIP サーバからキー・ファイルへのキーのコピー

データベース暗号化キーを KMIP サーバからキー・ファイルにコピーできます。これにより、バックアップ、およびネットワークや KMIP サービスの障害からのリカバリの両方でキーを利用できるようになります。以下を実行できます。

- KMIP サーバからのキーのコピーを使用して、データベース暗号化キー・ファイルを作成する
- データベース暗号化キーのコピーを KMIP サーバから既存のキー・ファイルに追加する

重要　暗号化キー・ファイルは必ず、しっかりと施錠された保管場所で管理されたリムーバブル・デバイスに保存してください。

KMIP サーバからのキーのコピーを使用したキー・ファイルの作成

キー・ファイルを作成して、そのキー・ファイルに KMIP サーバからキーをコピーするには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

```
>set $namespace="%SYS"
```

3. ^EncryptionKey を実行します。

```
%SYS>do ^EncryptionKey
```

4. ^EncryptionKey で、オプション [1]、[新規暗号化キー・ファイルの作成] を選択します。
5. 続いて表示されるプロンプトで、以下を指定します。
   - キー・ファイルの名前 (<install-dir>/mgr/ ディレクトリを基準にした相対名)。
   - キー・ファイルの説明。
   - キーの管理者の名前 – これは新しい管理者で、新しい名前を付けることができます。
   - その管理者のパスワード (および確認) – これは新しいパスワードで、有効な値を指定できます。
   - 利用可能な暗号セキュリティ・レベル – ファイルに保存されるキーの暗号化に使用するキーの長さ。
6. 次のプロンプトで、オプション [2]、[KMIP サーバからキーをコピー] を選択します。^EncryptionKey により、ファイルにコピーするキーを指定するよう求められます。
7. [キーの選択] プロンプトで、コピーするキーの番号を指定します。

^EncryptionKey により、指定した管理者のユーザ名とパスワードを使用してファイルが作成され、選択したキーがそのファイルに配置されます。

KMIP サーバから既存のキー・ファイルへのキーのコピーの追加

KMIP サーバから既存のキー・ファイルにキーを追加するには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、%SYS ネームスペースに移動します。

```
>set $namespace="%SYS"
```

3. ^EncryptionKey を実行します。

```
%SYS>do ^EncryptionKey
```

4. ^EncryptionKey で、オプション [2]、[既存の暗号化キー・ファイルの管理] を選択します。
5. [暗号化キー・ファイル] プロンプトで、キーを追加するキー・ファイルのパスと名前を入力します。パスは <install-dir>/mgr/ ディレクトリを基準にした相対パスです。
6. 次のプロンプトで、オプション [5]、[暗号化キーの追加] を選択します。続いて表示されるプロンプトで、以下を行います。
   a. [既存の管理者] で、キー・ファイルの管理者のユーザ名とパスワードを [ユーザ名] および [パスワード] に入力します。
   b. キー・ファイルに追加するキーの説明を入力します。
7. 次のプロンプトで、オプション [2]、[KMIP サーバからキーをコピー] を選択します。続いて表示されるプロンプトで、以下を行います。
   a. [KMIP サーバ] プロンプトで、キーのコピー元の KMIP サーバの名前を入力します。
   b. [キーの選択] プロンプトで、コピーするキーの番号を指定します。

^EncryptionKey により、選択したキーが選択したキー・ファイルに追加されます。

保存に依存しないキー管理タスク

一部のタスクは、ファイル内のキーおよび KMIP サーバ上のキーに対して行うタスクと同じです。

- データベース暗号化キーの無効化
- データ要素暗号化キーの無効化
- インスタンスの既定のデータベース暗号化キーまたはジャーナル暗号化キーの指定

データベース暗号化キーの無効化

データベース暗号化キーを無効にするには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、[**データベース暗号化**] ページ ([**システム管理**]→[**暗号化**]→[**データベース暗号化**]) に移動します。現在キーが有効になっている場合、キーの識別子はキーのテーブルに表示されます。
2. キーが新しい暗号化データベースまたはジャーナル・ファイルの暗号化の既定のキーである場合、そのキーを無効にすることはできません。これらいずれかのアクティビティに使用されているキーを無効にするには、それらに使用する別のキーを選択する必要があります。これを実行するには、別のキーに対して [**既定に設定**] または [**ジャーナル設定**] をクリックします。キーが前述のいずれかのアクティビティに使用されていない場合、キーの [**無効化**] ボタンが使用できるようになります。
3. キーを無効にするには、キーの行の [**無効化**] をクリックします。

   注釈　何らかの理由でキーを無効にできない場合、ポータルではエラー・メッセージが出力されます。以下の場合、キーを無効にできません。

   - IRISTEMP データベースおよび IRISLOCALDATA データベースが暗号化されている場合
   - このキーで暗号化され現在マウントされている (IRISTEMP および IRISLOCALDATA 以外の) 暗号化データベースがある場合
   - 現在、キーがジャーナル・ファイルの暗号化に使用されている場合(ジャーナル・ファイルの暗号化キーを変更する場合、ジャーナル・ファイルを切り替えるまで古い暗号化キーが引き続き使用されます)

   基盤となる状況への対処方法は、以下を参照してください。

4. 確認ダイアログで [OK] をクリックして、キーを無効にします。

キーを無効にするには、基盤となる状況に応じたアクションを実行する必要があります。

- IRISTEMP および IRISLOCALDATA 以外の暗号化データベースの場合、[**データベース**] ページ ([**システム処理**] > [**データベース**]) でデータベースをディスマウントします。これでキーを無効にできます。
- IRISTEMP および IRISLOCALDATA の場合、これらのデータベースが暗号化されないように指定して、InterSystems IRIS を再起動します。そのためには、[**データベース暗号化**] ページで [**起動設定の構成**] を選択します。起動時にデータベース暗号化キーを有効にしないように選択するか (この場合、InterSystems IRIS により IRISTEMP および IRISLOCALDATA の暗号化が無効にされます)、あるいは起動時にインタラクティブにまたは無人でデータベース暗号化キーを有効にするように選択できます (この場合、IRISTEMP および IRISLOCALDATA を暗号化するかどうかを選択できるので、[いいえ] を選択します)。
- 暗号化されたジャーナル・ファイルの場合、暗号化されたジャーナル・ファイルがリカバリに必要ないことを確認します。詳細は、"暗号化されたジャーナル・ファイル" を参照してください。

データ要素暗号化キーの無効化

データ要素暗号化キーを無効にするには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、[**データ要素暗号化**] ページ ([**システム管理**] > [**暗号化**] > [**データ要素暗号化**]) に移動します。有効になっているキーがある場合、ページにはそれらをリストするテーブルが表示されます。
2. 有効になっているキーのテーブルで、無効にするキーの [**無効化**] をクリックします。操作を確認するダイアログが表示されます。
3. 確認ダイアログで [OK] をクリックします。

[**データ要素暗号化**] ページが再び表示されると、無効化されたキーの行はテーブルに存在しなくなります。

インスタンスの既定のデータベース暗号化キーまたはジャーナル暗号化キーの指定

インスタンスにはそれぞれ既定のデータベース暗号化キーおよび既定のジャーナル暗号化キーがあります。インスタンスは、管理者がデータベース暗号化キーを初めてアクティブ化するときに、これらのそれぞれに初期値を設定します。最初に既定になるキーは、アクティブ化されたキー・ファイルにあるキーによって決まります。これらの値は、InterSystems IRIS をシャットダウンしてから再起動しても保持されます。

これらいずれかの目的で新しいキーを指定するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、[**データベース暗号化**] ページ ([**システム管理**] > [**暗号化**] > [**データベース暗号化**]) に移動します。現在有効なインスタンスの暗号化キーのテーブルが表示されます。
2. 暗号化キーのテーブルで、以下を実行します。
   - 新しい既定の暗号化キーを指定するには、そのキーに対して [**既定に設定**] をクリックします。現在の既定のキーに対する [**既定に設定**] ボタンは使用できません。
   - 新しいジャーナル暗号化キーを指定するには、そのキーに対して [**ジャーナル設定**] をクリックします。現在のジャーナル暗号化キーに対する [**ジャーナル設定**] ボタンは使用できません。
3. アクションの確認を求めるプロンプトが表示されたら、[OK] をクリックします。

InterSystems IRIS により、選択したキーが既定の暗号化キーまたはジャーナル暗号化キーとして設定されます。キーが既定の暗号化キーまたはジャーナル暗号化キーのいずれかである場合、そのキーは削除できません (InterSystems IRIS インスタンスの操作に必要なため)。したがって、キーをこれらのいずれかに指定すると、キーの [**削除**] ボタンは使用できなくなります。

暗号化データベースの使用法

暗号化データベースの使用法

機密情報が含まれるデータベース全体を保護するために、InterSystems IRIS® データ・プラットフォームではブロック・レベルのデータベース暗号化（または、データベース暗号化と略されています）がサポートされています。データベース暗号化は、エンティティ全体として暗号化されるデータベースの作成と管理ができるテクノロジです。InterSystems IRIS のキー管理ツールを使用して、これらのアクティビティをサポートします。

データベースの作成時に、これを暗号化することを選択できます。現在有効になっているキーがある場合にこのオプションを選択できます。暗号化データベースを作成すると、暗号化されていないデータベースと同様にこれを使用できます。暗号化テクノロジは透過的で、パフォーマンスに及ぼす影響は小さくて予測可能になります。

ここでは、暗号化データベースを作成および管理する方法を説明します。データベース暗号化機能では、監査ログとジャーナル・ファイルを暗号化する機能もサポートされています。これらの機能は両方とも、"暗号化の起動設定の構成"の説明に従って、起動時にデータベース暗号化キーにアクセスする必要があります。

暗号化データベースの作成

暗号化データベースを作成する場合は、データベースを新規作成するときに暗号化を指定します。ただし、暗号化データベースを作成する前に、InterSystems IRIS でデータベース暗号化キーを有効にしておく必要があります。以下はその方法です。

1. データベース暗号化キーを有効にします。
2. 管理ポータルのホーム・ページで、**[ローカルデータベース]** ページ（**[システム管理]** > **[構成]** > **[システム構成]** > **[ローカルデータベース]**）に移動します。
3. **[ローカルデータベース]** ページで、**[新規データベース作成]** を選択します。**[データベース]** ウィザードが表示されます。
4. ウィザードの 2 ページ目で、**[暗号化データベース?]** ボックスで **[はい]** を選択します。これによって、暗号化データベースが作成されます。ウィザードのその他すべてのページで、データベースを作成する際にデータベースに設定する特性を選択します（データベースの作成の詳細は、"ローカル・データベースの作成" を参照してください）。

注釈　InterSystems IRIS には、暗号化されていないデータベースの暗号化や暗号化データベースの解読のための暗号化管理ツールも必要に応じて用意されています。

暗号化データベースへのアクセスの確立

ミラーにデータベースを追加するなど、さまざまな操作を実行するには、データベースをマウントする必要があります。ただし、暗号化データベースをマウントする場合、そのキーを有効にする必要があります。したがって、データベースにアクセスするには、キーを有効にし、データベースをマウントする必要があります。その手順は以下のとおりです。

1. キーを有効にします。
2. 管理ポータルのホーム・ページで、**[データベース]** ページ（**[システム処理]** > **[データベース]**）に移動します。
3. このページで、マウントするデータベースに対して、データベースのテーブルでその行の右端の列にある **[マウント]** ボタンを選択します。確認画面で **[OK]** を選択すると、データベースがマウントされます。キーが有効になっていないと、データベースはマウントできず、エラー・メッセージが表示されます。

これで、データベースのデータにアクセスできるようになります。

暗号化データベースへの接続の切断

暗号化データベースへの接続を切断するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、[**データベース**] ページ ([**システム処理**] > [**データベース**]) に移動します。
2. このページで、データベースのテーブルで右にある [**ディスマウント**] ボタンを選択します。確認画面で [OK] を選択すると、データベースがディスマウントされます。
3. キーを無効にします。

データベースに対する読み取りおよび書き込みのたびに、有効になっているキーが使用されるので、先にデータベースをディスマウントしないとキーを無効にできません。データベースをディスマウントせずにキーを無効にしようとすると、エラー・メッセージが表示されます。

インスタンス間での暗号化データベースの移動

複数の InterSystems IRIS インスタンスを使用している組織では、あるインスタンスで異なるキーを使用して作成された暗号化データベースを別のインスタンスで使用することが必要になる場合があります。インスタンス間でデータを移動するには、データベースをバックアップした後、利用可能な暗号化管理ツールを使用してそのデータベースを再暗号化する必要があります。詳細は、"^EncryptionKey を使用したデータベース暗号化の変更" を参照してください。

暗号化の起動設定の構成

ここでは、データベース暗号化を行う 3 つの起動オプションそれぞれを設定する方法について説明します。

- キーを有効化しない起動 (既定) ー インスタンスには、起動時に利用可能なデータベース暗号化キーはありません。
- インタラクティブにキーを有効化する起動 ー インスタンスは、起動時にインタラクティブにデータベース暗号化キー情報を収集します。
- 無人のキー有効化による起動 ー インスタンスでは起動時に人的操作なしにデータベース暗号化キー情報を収集します。これは無人起動とも呼ばれます。

InterSystems IRIS のいくつかの機能では、起動時に (インタラクティブに、または無人起動によって) キーが利用可能である必要があります。

- InterSystems IRIS 監査ログの暗号化。
- **IRISTEMP** データベースおよび **IRISLOCALDATA** データベースの暗号化(両方とも暗号化するか、または両方とも暗号化しないかのいずれかです)。
- InterSystems IRIS ジャーナル・ファイルの暗号化。
- 暗号化データベースの起動時のマウント

キーを有効化しない起動

まだどのキーも有効化していない場合、これが InterSystems IRIS のインスタンスの既定の動作です。起動時のキーの有効化が設定されている場合に、これを解除するには、以下の手順に従います。

1. 管理ポータルのホーム・ページで、[**データベース暗号化**] ページ ([**システム管理**]→[**暗号化**]→[**データベース暗号化**]) に移動します。
2. [**起動設定の構成**] を選択します。InterSystems IRIS の起動の構成用オプション、および暗号化データベース用のその他のオプションがある領域が表示されます。
3. この領域で、[**起動オプション**] リストから [**なし**] を選択します。
4. [**保存**] をクリックします。以下の場合、このアクションを実行できない可能性があります。
   - 起動時に暗号化データベースが必要な場合。詳細は、"起動時に暗号化データベースが必要な場合" を参照してください。
   - 暗号化されたジャーナル・ファイルのいずれかでトランザクションが開いている場合。詳細は、"暗号化されたジャーナル・ファイル" を参照してください。

- 監査ログが暗号化されている場合(この場合、エラー・メッセージで暗号化データベースが参照されます。これは、監査ログが IRISAUDIT と呼ばれる InterSystems IRIS データベースに格納されているためです)。詳細は、"暗号化された監査ログ" を参照してください。

変更を妨げている問題に対処してから、この手順を再び実行してください。問題が修正されると、キーを有効化しない起動への変更を正常に行えるようになります。

起動時に暗号化データベースが必要な場合

起動時に必要な暗号化データベースがインスタンスにある場合、起動時にキーの有効化を含めないよう構成しようとすると、起動時に暗号化データベースが必要であること、およびキーの有効化オプションの変更が不可であることを示すエラー・メッセージが管理ポータルに表示されます(エラー・メッセージで IRISAUDIT データベースが参照されている場合、監査ログは暗号化されています)。

暗号化キーを有効化せずに InterSystems IRIS が起動するよう構成するために、起動後にのみ暗号化データベースをマウントできます。起動後にデータベースがマウントされるように構成するには、以下の手順に従います。

1. データベースがマウントされていることを確認し、そうでなければデータベースをマウントします。
   a. 管理ポータルのホーム・ページで、[**データベース**] ページ ([**システム処理**] > [**データベース**]) に移動します。
   b. データベースのテーブルで、そのデータベースの行を検索します。データベースがマウントされている場合、その行に [**ディスマウント**] オプションがあります。データベースがマウントされていない場合、[**ディスマウント**] オプションではなく、[**マウント**] オプションがあります。
   c. データベースがマウントされていない場合、[**マウント**] を選択します。
   d. 確認画面で [OK] を選択します(データベースは書き込み可能である必要があるため、[**読み取り専用**] チェック・ボックスにチェックを付けないでください)。
2. データベースが起動時にマウントされないよう、データベースのプロパティを編集します。
   a. [**ローカルデータベース**] ページ ([**システム管理**] > [**構成**] > [**システム構成**] > [**ローカルデータベース**]) に移動します。
   b. データベースのテーブルで、そのデータベースの行を検索します。
   c. データベース名をクリックしてデータベースを選択します。データベースを編集するためのページが表示されます。
   d. この [**編集**] ページで、[**起動時にマウントが必要**] チェック・ボックスのチェックを外します。
   e. [**保存**] をクリックします。

これでデータベースは起動時にマウントされなくなります。つまり、データベースは起動時にキーの有効化を必要としなくなります (他の理由で必要となる場合があります)。

暗号化されたジャーナル・ファイル

インスタンスでジャーナリングが使用されている場合、起動時にキーの有効化を含めないよう構成しようとすると、起動時にキーの有効化をオフにできない場合があります。該当する状況は以下のとおりです。

- そのジャーナル・ファイルを暗号化するようにインスタンスが構成されている場合
- 開いているトランザクションが (使用率の高いシステム上にある可能性が高い) ジャーナル・ファイルにある場合

この状況に該当する場合、起動時のキー有効化設定を変更する前に、暗号化されたジャーナル・ファイルの使用を中断する必要があります。そのための手順は以下のとおりです。

1. [**データベース暗号化**] ページ ([**システム管理**]→[**暗号化**]→[**データベース暗号化**]) で、[**暗号化ジャーナルファイル**] の設定を [**いいえ**] に変更します。[**起動時のキー有効化**] の設定はそのままにします。

2. ジャーナル・ファイルを切り替えます。これを行うには、**[ジャーナル]** ページ (**[システム処理]** > **[ジャーナル]**) で **[ジャーナル切り替え]** をクリックします。

暗号化されたジャーナル・ファイルで開いているトランザクションがすべてコミットまたはロールバックされると、InterSystems IRIS の起動の構成を変更できます。

注意　開いているトランザクションがなくなった後も、データベースをリストアする際に暗号化されたジャーナル・ファイルが必要になる場合があります。そのため、これらのファイルの暗号化に使用するキーがあるキー・ファイルのコピーを維持することが非常に重要です。

ジャーナル・ファイルの全般的な詳細は、"ジャーナリング" を参照してください。

暗号化された監査ログ

インスタンスに暗号化された監査ログがある場合、起動時にキーの有効化を含めないよう構成しようとすると、InterSystems IRIS では、起動時に暗号化データベースが必要であるという以下のようなエラー・メッセージが表示されます。

```
ERROR #1217: Can not disable database encryption key activation at startup.
Encrypted databases are required at startup:
C:\InterSystems\IRIS\Mgr\IRISAudit\
```

このエラー・メッセージでは暗号化データベースが参照されています。これは、監査ログが **IRISAUDIT** という InterSystems IRIS データベースに格納されているためです。

暗号化キーを有効化せずに InterSystems IRIS を起動すると、監査ログを暗号化できません。起動時にキーの有効化を含めないよう構成するには、InterSystems IRIS の設定を変更して、暗号化されていない監査ログをインスタンスで使用するように指定する必要があります。以下はその方法です。

1. インスタンスの監査データをバックアップします。
2. **[データベース暗号化]** ページ (**[システム管理]** > **[暗号化]** > **[データベース暗号化]**) に移動します。
3. **[起動設定の構成]** を選択すると、InterSystems IRIS の起動の構成用オプションおよび暗号化データベース用のその他のオプションがある領域が表示されます。
4. **[オプションで暗号化されたデータ]** の **[監査ログ暗号化]** リストで、**[いいえ]** をクリックします。

この設定を変更すると、既存の監査データがすべて消去され (ある場合)、直ちに暗号化されていない監査の使用が開始され、監査ログに AuditChange イベントが書き込まれます。

注意　監査データをバックアップしていない場合、監査ログの暗号化設定を変更すると該当する既存の監査データが失われます。

インタラクティブにキーを有効化する起動

キーが有効化されている場合、これが InterSystems IRIS のインスタンスの既定の動作です。インタラクティブなキーの有効化では、InterSystems IRIS のインスタンスにより、起動時にキーの場所およびキーの関連情報を入力するよう求められます。

重要　Windows の場合、インタラクティブなキーの有効化は、システム起動の一環で自動的に開始するサービスとして InterSystems IRIS を構成していると両立できなくなります。

インタラクティブなキーの有効化に対応するように InterSystems IRIS を構成するには、以下の操作を実行します。

1. 管理ポータルのホーム・ページで、**[データベース暗号化]** ページ (**[システム管理]**→**[暗号化]**→**[データベース暗号化]**) に移動します。
2. **[起動設定の構成]** を選択します。**[起動オプション]** 領域が表示され、**[起動時のキー有効化]** リストが表示されます。
3. **[起動時のキー有効化]** リストで、**[インタラクティブ]** を選択します。それまでのこのフィールドの値が **[なし]** となっていた場合は、この操作によってページの **[オプションで暗号化されたデータ]** 領域が表示されます。

4. この領域のフィールドは以下のとおりです。

   - **[IRISTEMP および IRISLOCALDATA データベースの暗号化]** – IRISTEMP データベースと IRISLOCALDATA データベースを暗号化するかどうかを指定できます。暗号化する場合は **はい** を選択し、暗号化しない場合は **[いいえ]** を選択します。
   - **[暗号化ジャーナルファイル]** – インスタンスで独自のジャーナル・ファイルを暗号化するかどうかを指定できます。ジャーナル・ファイルを暗号化する場合は **[はい]** を選択し、暗号化しない場合は **[いいえ]** を選択します。InterSystems IRIS を起動すると新しいジャーナル・ファイルが作成されるため、選択は起動オプションによって決まります。暗号化を選択した場合は、起動時にキーが必要です。

     注釈　この変更は、InterSystems IRIS が次にジャーナル・ファイルを切り替えたときに有効になります。再起動せずにジャーナル・ファイルの暗号化を開始するには、このページの操作を完了してからジャーナル・ファイルを切り替えます。

   - **[監査ログ暗号化]** – InterSystems IRIS が監査ログを暗号化するかどうかを指定できます。監査ログを暗号化する場合は **[はい]** を選択し、暗号化しない場合は **[いいえ]** を選択します。InterSystems IRIS を起動するとさまざまなイベントが監査ログに記録されるため、選択は起動オプションによって決まります。暗号化を選択した場合は、起動時にキーが必要です。

     注意　この変更は直ちに反映され、既存の監査データがすべて削除されます。この設定を変更する前に監査データベースをバックアップしてください。そうしないと、監査データが失われます。

5. **[保存]** をクリックして、選択した設定内容を保存します。

重要　InterSystems IRIS が以下のように構成されている場合を考えます。

- **IRISTEMP** および **IRISLOCALDATA**、ジャーナル・ファイル、または監査ログを暗号化する
- 起動時に暗号化データベースを必要とする

この場合、必要な暗号化キーを有効にしないと InterSystems IRIS の起動に失敗します。起動に失敗した場合は、InterSystems IRIS の緊急起動モードを使用して、起動時に暗号化機能をまったく必要としないように InterSystems IRIS を構成します。

無人のキー有効化による起動

無人でキーを有効化する起動は無人起動とも呼ばれ、キーを有効化し、可能であれば起動時に人的操作なしに暗号化データベースをマウントします。無人起動を正常に実行するには、インスタンスが以下にアクセスできる必要があります。

- 暗号化データベース。
- データベース暗号化キー。以下のいずれかの方法によります。
  - キーを格納する KMIP サーバ
  - データベース暗号化キー・ファイル（キー、およびユーザ名とパスワード（データベース暗号化キーの無人有効化に使用されます）が格納されます）

このセクションでは、以下のトピックについて説明します。

- KMIP サーバ上のキーを使用した無人起動の構成
- キー・ファイル内のキーを使用した無人起動の構成

- 無人起動の問題への一時的な対処

注意　これらの要素をすべて利用可能にすることで、InterSystems IRIS にあるデータのセキュリティは、これらの要素が保持されているマシンの物理的セキュリティに、全面的に依存することになります。このような物理的セキュリティが確保されていないサイトのデータは、そのデータが暗号化されていない場合と同等のセキュリティ上のリスクにさらされます。この状態を避けるには、インタラクティブな起動を使用するか(こうすれば、すべての要素が同時にリスクにさらされることがなくなります)、関連するマシンの物理的セキュリティを確保するようにします。

KMIP サーバ上のキーを使用した無人起動の構成

KMIP サーバ上のキーを使用して無人で起動するよう InterSystems IRIS インスタンスを構成するには、以下の手順に従います。

1. 関連するインスタンスに対して、ターミナルを起動し、十分な特権を持つユーザとしてログインします。
2. ターミナル・プロンプトで、**%SYS** ネームスペースに移動します。

   ```
   >set $namespace="%SYS"
   ```

3. ^EncryptionKey を実行します。

   ```
   %SYS>do ^EncryptionKey
   ```

4. ^EncryptionKey で、オプション **[3]**、**[データベース暗号化]** を選択します。
5. 次のプロンプトで、オプション **[4]**、**[起動オプションの構成]** を選択します。
6. 次のプロンプトで、オプション **[4]**、**[KMIP サーバを使用したキーの無人有効化]** を選択します。
7. **[KMIP サーバのインスタンス名]** プロンプトで、KMIP サーバ構成の名前を入力します。
8. 続いて表示されるプロンプトで、暗号化する項目を指定します(これらすべての項目で起動時に有効なキーが必要になります)。
   - **[暗号化ジャーナルファイル]** (既定は [いいえ]) ― インスタンスで独自のジャーナル・ファイルを暗号化するかどうかを指定できます。ジャーナル・ファイルを暗号化する場合は **[はい]** を入力し、暗号化しない場合は **[いいえ]** (既定) を入力または選択します。InterSystems IRIS を起動すると新しいジャーナル・ファイルが作成されるため、選択は起動オプションによって決まります。暗号化を選択した場合は、起動時にキーが必要です。

     この変更は、InterSystems IRIS が次にジャーナル・ファイルを切り替えたときに有効になります。既定では、これは InterSystems IRIS を次回再起動したときに実行されます。再起動せずにジャーナル・ファイルの暗号化を開始するには、このページの操作を完了してからジャーナル・ファイルを切り替えます。
   - **[IRISTEMP および IRISLOCALDATA データベースの暗号化]** (既定は [いいえ]) ― IRISTEMP データベースと IRISLOCALDATA データベースを暗号化するかどうかを指定できます。暗号化する場合は **[はい]** を入力し、暗号化しない場合は **[いいえ]** (既定) を入力または選択します。
   - **[監査データベースの暗号化]** (既定は [いいえ]) ― InterSystems IRIS が監査ログを暗号化するかどうかを指定できます。監査ログを暗号化する場合は **[はい]** を選択し、暗号化しない場合は **[いいえ]** (既定) を選択します。InterSystems IRIS を起動するとさまざまなイベントが監査ログに記録されるため、選択は起動オプションによって決まります。暗号化を選択した場合は、起動時にキーが必要です。

     注意　この変更は直ちに反映され、既存の監査データがすべて削除されます。この設定を変更する前に監査データベースをバックアップしてください。そうしないと、監査データが失われます。
9. 続いて、起動時に有効化する KMIP キーの現在のリストが表示され、次のアクションを指定するよう求められます。
   - 起動キーのリストにキーを追加するには、オプション **[1]**、**[キーをリストに追加]** を選択します。

- 起動キーのリストからキーを削除するには、オプション [2]、[**キーをリストから削除**] を選択します。
- 起動キーのリストを保存するには、オプション [3]、[**リストの保存**] を選択します。

10. リストに、起動時に有効化する KMIP キーの目的のリストが含まれる場合、オプション [3] を選択して、リストを保存します。

キー・ファイル内のキーを使用した無人起動の構成

注意　無人起動するように InterSystems IRIS を構成すると、そのインスタンスによってデータベース暗号化キー・ファイルに別の管理者が追加されます。その管理者にはシステムによって生成された名前とパスワードがあります。InterSystems IRIS によってキー・ファイルが変更されてこのユーザ名とパスワードが追加された後は、鍵付きのラック設置式 CD-ROM ドライブまたは DVD ドライブのように物理的に鍵がかけられるハードウェア上にのみキー・ファイルのコピーを配置することを強くお勧めします。さらに、このハードウェアが格納されているデータ・センタの施設を施錠して監視下に置く必要があります。データベース暗号化キーは、そのキーを使用して暗号化したデータベースと同じドライブには格納しないでください。

キー・ファイル内のキーを使用して無人で起動するよう InterSystems IRIS インスタンスを構成するには、以下の手順に従います。

1. 先にキーを有効にしておく必要があります。キーを有効にするには、"キーの有効化" を参照してください。
2. 管理ポータルのホーム・ページで、[**データベース暗号化**] ページ ([**システム管理**]→[**暗号化**]→[**データベース暗号化**]) に移動します。
3. [**起動設定の構成**] を選択します。[**起動オプション**] リストが表示されます。
4. [**起動オプション**] で、[**自動 (推奨されていません)**] を選択します。これにより、ページに表示されるフィールドが変更されます。
5. [**起動オプション**] 領域が展開され、3 つのフィールドが表示されます。以下を設定します。
   - [**キー・ファイル**] – データベース暗号化キー・ファイルのパス。ここでは絶対パスと相対パスのどちらでも指定できます。相対パスで指定した場合は、InterSystems IRIS のインストール・ディレクトリが基準になります。[**参照**] をクリックすると、ファイル・システム上でデータベース暗号化キー・ファイルを探します。
   - [**管理者名**] – このキー・ファイルの管理者。
   - [**パスワード**] – 管理者のパスワード。
6. [**オプションで暗号化されたデータ**] 領域で、以下のフィールドにすべて入力します。
   - [**IRISTEMP および IRISLOCALDATA データベースの暗号化**] – IRISTEMP データベースと IRISLOCALDATA データベースを暗号化するかどうかを指定できます。暗号化する場合は **はい** を選択し、暗号化しない場合は [**いいえ**] を選択します。
   - [**暗号化ジャーナルファイル**] – インスタンスで独自のジャーナル・ファイルを暗号化するかどうかを指定できます。ジャーナル・ファイルを暗号化する場合は [**はい**] を選択し、暗号化しない場合は [**いいえ**] を選択します。InterSystems IRIS を起動すると新しいジャーナル・ファイルが作成されるため、選択は起動オプションによって決まります。暗号化を選択した場合は、起動時にキーが必要です。

     注釈　この変更は、InterSystems IRIS が次にジャーナル・ファイルを切り替えたときに有効になります。既定では、これは InterSystems IRIS を次回再起動したときに実行されます。再起動せずにジャーナル・ファイルの暗号化を開始するには、このページの操作を完了してからジャーナル・ファイルを切り替えます。

   - [**監査ログ暗号化**] – InterSystems IRIS が監査ログを暗号化するかどうかを指定できます。監査ログを暗号化する場合は [**はい**] を選択し、暗号化しない場合は [**いいえ**] を選択します。InterSystems IRIS を起動するとさまざ

まなイベントが監査ログに記録されるため、選択は起動オプションによって決まります。暗号化を選択した場合は、起動時にキーが必要です。

注意　この変更は直ちに反映され、既存の監査データがすべて削除されます。この設定を変更する前に監査データベースをバックアップしてください。そうしないと、監査データが失われます。

7. [保存] をクリックして、選択した設定内容を保存します。

無人起動の問題への一時的な対処

InterSystems IRIS が以下のように構成されている場合を考えます。

- IRISTEMP および IRISLOCALDATA、ジャーナル・ファイル、または監査ログを暗号化する
- 起動時に暗号化データベースを必要とする

この場合、暗号化キーを有効にしないと InterSystems IRIS の起動に失敗します。起動に失敗した場合は、InterSystems IRIS の緊急起動モードを使用して、起動時に暗号化機能をまったく必要としないように InterSystems IRIS を構成します。

InterSystems IRIS 付属のデータベースの暗号化

InterSystems IRIS の各インスタンスには、数多くのデータベースが付属しています。暗号化する機能および暗号化の値は、以下のようにデータベースによって異なります。

- IRISLOCALDATA：IRISTEMP データベースと組み合わせて暗号化できます。IRISLOCALDATA を暗号化するには、起動時にこのデータベースを必要とするため、キーが起動時に使用可能である必要があります。
- IRISAUDIT：暗号化できます。IRISAUDIT を暗号化するには、起動時にこのデータベースを必要とするため、キーが起動時に使用可能である必要があります。
- IRISLIB：暗号化しないでください。(IRISLIB の内容はすべて公開されます。)
- IRISSYS：暗号化しないでください。インスタンスに含まれているこのデータベースが暗号化された形式になっている場合、InterSystems IRIS は起動できません。
- IRISTEMP：IRISLOCALDATA データベースと組み合わせて暗号化できます。IRISTEMP を暗号化するには、起動時にこのデータベースを必要とするため、キーが起動時に使用可能である必要があります。
- USER：暗号化できます。

^EncryptionKey を使用したデータベース暗号化の変更

管理ポータルからは実行できない暗号化管理操作の実行が必要になる場合があります。^EncryptionKey ユーティリティを使用すると、以下のアクションを実行できます。

- 暗号化されていないデータベースを暗号化データベースに変換する
- 暗号化データベースを暗号化されていないデータベースに変換する
- 新しいキーを使用するように暗号化データベースを変換する

^EncryptionKey ユーティリティで使用されるツールには以下のことが当てはまります。

^EncryptionKey ユーティリティでは、一連の暗号化管理ツールが使用されます。

- 暗号化関連アクティビティで組み込みのハードウェア命令が利用可能な場合、これらのアクティビティは、ソフトウェア・ベースの暗号化よりもかなり高速です。暗号化管理ツールは、利用可能であればハードウェア命令を使用します。
- 暗号化管理ツールは、KMIP サーバ上に保存されたキーを使用できます。

- 暗号化管理ツールは FIPS モードで実行できます。

注釈　暗号化管理ツールはジャーナル・ファイルでは動作しません。

暗号化されていないデータベースを暗号化データベースに変換する

暗号化されていないデータベースを暗号化データベースに変換するには、以下の手順に従います。

1. 暗号化されるデータベースのデータをバックアップします。

   InterSystems IRIS は、所定位置のデータを暗号化します。つまり、この処理ではディスク上の領域を使用します (データベースを別の場所にコピーして正常解読後に現在のディスクの場所にデータベースをリストアする処理は行いません)。処理が完了する前にユーティリティが無効になった場合、データベースは暗号化された部分と暗号化されていない部分が混在して、使用不可能になります。

   注意　データベースを変換する前にそのバックアップを作成しておくことは重要です。バックアップを作成しておかないと、データが失われる可能性があります。

2. データベースを暗号化するキーをキー・ファイルまたは KMIP サーバから有効化します。
3. ターミナルを開始します。
4. %SYS ネームスペースで、^EncryptionKey ユーティリティを実行します。
5. ^EncryptionKey で、オプション [3]、[**データベース暗号化**] を選択します。
6. [データベース暗号化] サブメニューで、オプション [7]、[**既存データベースの暗号化ステータスの変更**] を選択します。
7. [**データベースディレクトリ**] サブメニューで、変更するデータベースを選択します。データベースがディレクトリ別にリストされます。データベースを選択すると、データベースが暗号化されているかどうかが通知されます。
8. データベースが暗号化されていない場合、このルーチンで暗号化できます。[**データベースを暗号化しますか?**] プロンプトで、yes または y を入力します。大文字と小文字は区別されません。
9. [**暗号化のキーの選択**] プロンプトで、データベースの暗号化に使用するキーを選択します。データベースが現在マウントされている場合は、その情報が表示されます。
10. データベースが現在マウントされている場合は、その情報が表示されます。[**データベースディスマウント**] プロンプトで、yes または y を入力します。大文字と小文字は区別されません。

    重要　データベースをディスマウントしてから再マウントすると操作が中断されるため、適切な予防措置を実施して、問題が発生しないようにしてください。

続いて、ルーチンによってデータベースが暗号化されます。このプロセスの一部として、データベースがマウントされていた場合、データベースをディスマウントしてマウントしたことを示すメッセージが表示されます。データベースが再びマウントされたら、暗号化は完了です。

暗号化データベースを暗号化されていないデータベースに変換する

暗号化データベースを暗号化されていないデータベースに変換するには、以下の手順に従います。

1. 暗号化されないデータベースのデータをバックアップします。

   InterSystems IRIS は、所定位置のデータの暗号化を解除します。つまり、この処理ではディスク上の領域を使用します (データベースを別の場所にコピーして正常解読後に現在のディスクの場所にデータベースをリストアする処理

は行いません)。処理が完了する前にユーティリティが無効になった場合、データベースは暗号化された部分と暗号化されていない部分が混在して、使用不可能になります。

注意　データベースを変換する前にそのバックアップを作成しておくことは重要です。バックアップを作成しておかないと、データが失われる可能性があります。

2. データベースを暗号化するキーをキー・ファイルまたは KMIP サーバから有効化します。
3. ターミナルを開始します。
4. %SYS ネームスペースで、^EncryptionKey ユーティリティを実行します。
5. ^EncryptionKey で、オプション [3]、[**データベース暗号化**] を選択します。
6. [データベース暗号化] サブメニューで、オプション [7]、[**既存データベースの暗号化ステータスの変更**] を選択します。
7. [**データベースディレクトリ**] サブメニューで、変更するデータベースを選択します。データベースがディレクトリ別にリストされます。データベースを選択すると、データベースが暗号化されているかどうかが通知されます。データベースが暗号化されていて、その暗号化キーが有効化されていない場合、その旨も通知されます。
8. データベースが暗号化されている場合、このルーチンで解読できます。[**データベースを解読しますか?**] プロンプトで、yes または y を入力します。大文字と小文字は区別されません。
9. データベースの暗号キーがレポートされた後、データベースを別のキーで暗号化するかどうかを尋ねられます。Enter キーを押して、データベースを解読済みデータベースに変換し、新しいキーを使用して暗号化します。
10. データベースが現在マウントされている場合は、その情報が表示されます。[**データベースディスマウント**] プロンプトで、yes または y を入力します。大文字と小文字は区別されません。

重要　データベースをディスマウントしてから再マウントすると操作が中断されるため、適切な予防措置を実施して、問題が発生しないようにしてください。

続いて、ルーチンによってデータベースが解読されます。このプロセスの一部として、データベースがマウントされていた場合、データベースをディスマウントしてマウントしたことを示すメッセージが表示されます。データベースが再びマウントされたら、解読は完了です。

新しいキーを使用するように暗号化データベースを変換する

新しいキーを使用するように暗号化データベースを変換するには、以下の手順に従います。

1. 再暗号化されるデータベースのデータをバックアップします。

   InterSystems IRIS は、所定位置のデータを暗号化します。つまり、この処理ではディスク上の領域を使用します(データベースを別の場所にコピーして正常解読後に現在のディスクの場所にデータベースをリストアする処理は行いません)。処理が完了する前にユーティリティが無効になった場合、データベースは暗号化された部分と暗号化されていない部分が混在して、使用不可能になります。

   注意　データベースを変換する前にそのバックアップを作成しておくことは重要です。バックアップを作成しておかないと、データが失われる可能性があります。

2. キー・ファイルまたは KMIP サーバから、データベースを暗号化するキーと再暗号化するキーを有効化します。
3. ターミナルを開始します。
4. %SYS ネームスペースで、^EncryptionKey ユーティリティを実行します。
5. ^EncryptionKey で、オプション [3]、[**データベース暗号化**] を選択します。
6. [データベース暗号化] サブメニューで、オプション [7]、[**既存データベースの暗号化ステータスの変更**] を選択します。

7. [**データベースディレクトリ**] サブメニューで、変更するデータベースを選択します。データベースがディレクトリ別にリストされます。データベースを選択すると、データベースが暗号化されているかどうかが通知されます。
8. データベースが暗号化されている場合、このルーチンで解読できます。[**データベースを解読しますか?**] プロンプトで、yes または y を入力します。大文字と小文字は区別されません。
9. 次のプロンプト [**データベースを再暗号化しますか?**] で、yes または y を入力します。大文字と小文字は区別されません。
10. [**暗号化のキーの選択**] プロンプトで、データベースの暗号化に使用するキーを選択します。
11. データベースが現在マウントされている場合は、その情報が表示されます。[**データベースディスマウント**] プロンプトで、yes または y を入力します。大文字と小文字は区別されません。

    重要　　データベースをディスマウントしてから再マウントすると操作が中断されるため、適切な予防措置を実施して、問題が発生しないようにしてください。

続いて、ルーチンによってデータベースが再暗号化されます。このプロセスの一部として、データベースがマウントされていた場合、データベースをディスマウントしてマウントしたことを示すメッセージが表示されます。データベースが再びマウントされたら、暗号化は完了です。

データ要素暗号化

データ要素暗号化の使用法

データ要素暗号化により、データベース全体を暗号化するよりも、アプリケーション・データをよりきめ細かく暗号化する方法が実現されます。漏洩防止が必要な機密データ要素のためのものです。例えば、顧客のレコードでクレジット・カード情報のフィールドを排他的に暗号化できます。検査結果 (HIV 検査など) を表示するフィールドを患者レコードで排他的に暗号化できます。社会保障番号が記載されたレコードで該当フィールドを排他的に暗号化できます。

データ要素暗号化は、管理ポータルからではなく、(API により) プログラムで使用できます。API によりアクセスできるので、アプリケーション・コードでこれを使用できます。データ要素暗号化をデータベース暗号化で使用するオプションがあります (両方を使用する必要はありません)。

アプリケーションでデータ要素暗号化を使用するには、アプリケーション実行時に必要なキーが使用可能である必要があります。キーを使用可能にするには、これを有効にします。詳細は、"プログラムで管理するキー"、またはポータルを使用する場合は "データ要素暗号化キーの有効化" を参照してください。キーが有効化されると、InterSystems IRIS® データ・プラットフォームではその一意の識別子が有効化キーのテーブルに表示されます。アプリケーションではその識別子を使用してキーを参照し、暗号化処理のためにメモリにロードできます。同時に 4 つまでのキーを有効にできるので、データ要素暗号化では複数のキーが必要なタスクのインフラストラクチャが実現します。

データ要素暗号化のデータを暗号化する際、InterSystems IRIS では結果として得られる暗号化テキストと共に暗号化キーの一意の識別子を格納します。一意の識別子により、暗号化テキスト自体のみを使用して解読時にシステムでキーを識別できます。

ここでは以下について説明します。

- プログラムで管理するキー
- データ要素暗号化の呼び出し
- リアルタイムのデータの再暗号化のサポート

プログラムで管理するキー

データ要素暗号化は API により使用できるので、キーを管理するための一連の呼び出しもあります。

- $SYSTEM.Encryption.CreateEncryptionKey
- $SYSTEM.Encryption.ActivateEncryptionKey
- $SYSTEM.Encryption.DeactivateEncryptionKey
- $SYSTEM.Encryption.ListEncryptionKeys

これらはすべて、%SYSTEM.Encryption クラスのメソッドです。

データ要素暗号化の呼び出し

データ要素暗号化で利用できるシステム・メソッドは、%SYSTEM.Encryption クラスのすべてのメソッドで、以下のとおりです。

- $SYSTEM.Encryption.AESCBCManagedKeyEncrypt
- $SYSTEM.Encryption.AESCBCManagedKeyDecrypt
- $SYSTEM.Encryption.AESCBCManagedKeyEncryptStream
- $SYSTEM.Encryption.AESCBCManagedKeyDecryptStream

上記のメソッド名の先頭はすべて“AESCBCManagedKey”になります。それらのメソッドが AES (Advanced Encryption Standard) を CBC (Cipher Block Chaining) モードで使用し、マネージド・キー暗号化の一連のツールの一部であるためです。

重要　“ManagedKey”が名前に含まれない AESCBC メソッドは、古いメソッドでこれらの目的では使用できません。

$SYSTEM.Encryption.AESCBCManagedKeyEncrypt

このメソッドのシグニチャは、通常の呼び出しと同様に以下のようになります。

```
$SYSTEM.Encryption.AESCBCManagedKeyEncrypt
        (
        plaintext As %String,
        keyID As %String,
        )
    As %String
```

各要素の内容は以下のとおりです。

- plaintext ー 暗号化対象となる暗号化されていないテキスト。
- keyID ー 平文の暗号化に使用されるデータ暗号化キーの GUID。
- メソッドは、暗号化されたテキストを返します。

メソッドが正常に実行できなかった場合、このメソッドは <FUNCTION> エラーまたは <ILLEGAL VALUE> エラーをスローします。このメソッドの呼び出しを Try-Catch ループに記述します。Try-Catch の詳細は、"TRY-CATCH メカニズム" を参照してください。

詳細は、"$SYSTEM.Encryption.AESCBCManagedKeyEncrypt" のクラス・リファレンス・コンテンツを参照してください。

$SYSTEM.Encryption.AESCBCManagedKeyDecrypt

このメソッドのシグニチャは、通常の呼び出しと同様に以下のようになります。

```
$SYSTEM.Encryption.AESCBCManagedKeyDecrypt
        (
        ciphertext As %String
        )
    As %String
```

各要素の内容は以下のとおりです。

- ciphertext ー 解読対象となる暗号化テキスト。
- メソッドは、解読された平文を返します。

メソッドが正常に実行できなかった場合、このメソッドは <FUNCTION> エラーまたは <ILLEGAL VALUE> エラーをスローします。このメソッドの呼び出しを Try-Catch ループに記述します。Try-Catch の詳細は、"TRY-CATCH メカニズム" を参照してください。

キー ID は解読対象となる暗号化テキストに関連付けられるため、キー ID をこの呼び出しで含める必要はありません。

詳細は、"$SYSTEM.Encryption.AESCBCManagedKeyDecrypt" のクラス・リファレンス・コンテンツを参照してください。

$SYSTEM.Encryption.AESCBCManagedKeyEncryptStream

このメソッドのシグニチャは、通常の呼び出しと同様に以下のようになります。

```
$SYSTEM.Encryption.AESCBCManagedKeyEncryptStream
        (
        plaintext As %Stream.Object,
        ciphertext As %Stream.Object,
        keyID As %String,
        )
    As %Status
```

各要素の内容は以下のとおりです。

- plaintext ー 暗号化対象となる暗号化されていないストリーム。
- ciphertext ー 暗号化ストリームを受け取る変数。
- keyID ー plaintext の暗号化に使用されるデータ暗号化キーの GUID。
- メソッドは %Status コードを返します。

詳細は、"$SYSTEM.Encryption.AESCBCManagedKeyEncryptStream" のクラス・リファレンス・コンテンツを参照してください。

$SYSTEM.Encryption.AESCBCManagedKeyDecryptStream

このメソッドのシグニチャは、通常の呼び出しと同様に以下のようになります。

```
$SYSTEM.Encryption.AESCBCManagedKeyDecryptStream
        (
        ciphertext As %Stream.Object,
        plaintext As %Stream.Object
        )
    As %Status
```

各要素の内容は以下のとおりです。

- ciphertext ー 解読対象となる暗号化ストリーム。
- plaintext ー 解読ストリームを受け取る変数。
- メソッドは %Status コードを返します。

キー ID は解読対象となる暗号化テキストに関連付けられるため、キー ID をこの呼び出しで含める必要はありません。

詳細は、"$SYSTEM.Encryption.AESCBCManagedKeyDecryptStream" のクラス・リファレンス・コンテンツを参照してください。

リアルタイムのデータの再暗号化のサポート

データ要素暗号化では、InterSystems IRIS アプリケーションで、暗号化されたデータ要素を新しいキーで再暗号化する処理をサポートできます。

暗号化されたデータ要素には暗号化キーの識別子が格納されているので、これによって、データの再暗号化の処理が単純になります。暗号化テキストのハンドルと有効なキーを与えられるだけで、アプリケーションは再暗号化を実行できます。例えば、データ要素暗号化では、機密データをダウンタイムなしで再暗号化できる機能がサポートされています。これは特に、PCI DSS (Payment Card Industry Data Security Standard) 要件を満たすことが必要な場合など、法的な理由でこの処理が必要な場合に便利です。

データの再暗号化が必要な場合、新しいキーを作成し、アプリケーションに対して、これが新しい暗号化キーであることを指定します。その後、要素を解読してから新しいキーで暗号化するバックグラウンド・アプリケーションを実行するなどの処理を実行できます。この処理では暗号化に指定されたキーを使用し、解読にも必ずその適切なキーを使用します。これはキーが暗号化されたデータと共に格納されるためです。

データ損失に対する保護

データ損失に対する保護

暗号化データをいつでも利用できるようにするために、インターシステムズでは、以下の予防措置を取ることを強くお勧めします。

- キー・ファイルを使用している場合、使用している各ファイルに対して、以下の手順に従います。
  1. キー・ファイルの追加の管理者を作成します。
  2. その管理者のユーザ名とパスワードを紙に記録します。
  3. 記録したユーザ名とパスワードを、キーの使用場所から十分に離れたところにある耐火金庫などの物理的に安全な場所に配置します。
  4. キー・ファイルのバックアップ・コピーを作成し、記録したユーザ名とパスワードと同じ安全な場所に配置します。
- KMIP サーバを使用している場合は、サーバ・ベンダの指示に従って、そのサーバの内容をバックアップします。

注意　これらの予防措置を怠ると、暗号化データが永久にアクセス不能となり、そのデータをまったく読み取れなくなる場合があります。

緊急事態への対処

緊急事態への対処

暗号化データが関係する緊急事態が生じた場合、その暗号化データへのアクセスが永久に失われることになる可能性があります。緊急事態が生じた場合は、データが永久に失われるリスクを最小限に抑えるために即座に対処する必要があります。

ここでは、暗号化データが関係する緊急事態が発生した場合に取るべき手順について説明します。緊急事態に対して予防措置を取るには、"データ損失に対する保護" を参照してください。

キー・ファイル使用時における緊急事態への対処

ここでは、データを損失する恐れがある事態において、どのように対処するべきかについて説明します。これらの状況には以下が含まれます。

- 有効なキーが保存されているファイルが損傷したり紛失した場合
  - 既知の管理者のユーザ名とパスワードのあるキー・ファイルのバックアップ・コピーがある場合
  - キー・ファイルのバックアップ・コピーがない場合、またはキーに既知の管理者のユーザ名とパスワードがない場合

    注意　これは緊急事態です。即座に対処してください。

- 起動時に必要なデータベース暗号化キー・ファイルが存在しない場合
  - キー・ファイルを使用可能にできる場合
  - バックアップ・キー・ファイルが使用可能な場合
  - キー・ファイルが使用できない場合

有効なキーが保存されているファイルが損傷したり紛失した場合

この場合、以下の状況が発生します。

- データベース暗号化キーが InterSystems IRIS® データ・プラットフォーム・インスタンスに対して有効化されています。
- InterSystems IRIS が暗号化データを使用します。
- データベース暗号化キーが保存されているキー・ファイルが破損します。

既知の管理者のユーザ名とパスワードのあるキー・ファイルのバックアップ・コピーがある場合

注意　この手順は、InterSystems IRIS データベースの暗号化データが失われる恐れがある緊急事態に実行します。

有効なキーが保存されているファイルがアクセス不能になった場合や損傷した場合は、直ちに以下の手順を実行します。

1. キー・ファイルのバックアップ・コピーを取得します。これは、"暗号化データのアクセスにおける偶発的な損失からの保護" で説明されている手順に従って保存したコピーです。
2. キー・ファイルの新しいバックアップ・コピーを作成して、安全な場所に保存します。
3. キーの新しいコピーが使用されるように InterSystems IRIS を設定します。

- インタラクティブな起動を使用している場合、スタートアップ・プロシージャにキーの新しいコピーを組み込みます。
- 無人起動を使用している場合、以前と同じ起動オプションに設定している場合でも、キー・ファイルの新しいコピーで起動オプションを再構成します。

キー・ファイルのバックアップ・コピーがない場合、またはキーに既知の管理者のユーザ名とパスワードがない場合

警告　この手順は、InterSystems IRIS データベースの暗号化データが失われる恐れがある緊急事態に実行します。

有効なキーが保存されているファイルが InterSystems IRIS の実行中にアクセス不能になった場合や損傷した場合は、そのキーで暗号化されている各データベースに対し、直ちに以下の手順を実行します。

1. 警告　InterSystems IRIS をシャットダウンしたり有効になっているキーを無効にしたりすると、データが永久に失われます。

   InterSystems IRIS をシャットダウンしないでください。

   現在有効になっているキーは無効にしないでください。

2. インターシステムズのサポート窓口までお問い合わせください。サポート窓口のエンジニアが以下の手順をご案内して、あらゆるご質問にお答えします。
3. データベースをディスマウントします。これにより、暗号化されていないデータベースにデータをコピーしている最中に、暗号化されたコンテンツがあるデータベースではどのユーザも変更ができないようにします。
   a. 管理ポータルのホーム・ページで、[**データベース**] ページ ([**システム処理**] > [**データベース**]) に移動します。
   b. 暗号化データベースがマウントされている場合、[**データベース**] ページで、データベースの行にある最後から 2 番目の列で [**ディスマウント**] オプションを選択します。確認ダイアログで [OK] をクリックします。
   c. [**データベース**] ページが再び表示されたら、データベースの行にある最後の列で [**マウント**] オプションを選択します。
   d. [**データベースマウント**] 確認画面で、[**読み取り専用**] チェック・ボックスにチェックを付け、[OK] を選択します。

   この手順の間、誰もデータベースに対して変更を行わないことが重要です。データベースを読み取り専用でマウントすると、ユーザはデータを変更できなくなります。
4. 暗号化されていない状態ですべてのデータを別のデータベースにコピーします。データをコピーする手順は以下のとおりです。
   a. ターミナルで、%SYS ネームスペースに移動します。

```
REGULARNAMESPACE>set $namespace="%SYS"
```

   b. そのネームスペースで ^GBLOCKCOPY コマンドを実行します。

```
%SYS>d ^GBLOCKCOPY

This routine will do a fast global copy from a database to another database or
to a namespace. If a namespace is the destination, the global will follow any
mappings set up for the namespace.

1) Interactive copy
2) Batch copy
3) Exit

Option?1
```

c. ^GBLOCKCOPY プロンプトで、インタラクティブ・コピーとして 1 を指定します。

```
Option? 1

1) Copy from Database to Database
2) Copy from Database to Namespace
3) Exit

Option?
```

d. ^GBLOCKCOPY プロンプトで、コピー・タイプの入力が要求されたら、別のデータベースへのコピーを指定する 1 を選択します。

```
Option? 1
Source Directory for Copy (? for List)?
```

ここで、暗号化データベースの名前を指定するか、? と入力して番号付けされたデータベースのリストを表示します。リストには暗号化データベースが含まれます。? と入力すると、^GBLOCKCOPY は以下のようなリストを表示します。

```
Source Directory for Copy (? for List)? ?

1) C:\InterSystems\MyIRIS\mgr\
2) C:\InterSystems\MyIRIS\mgr\irislocaldata\
3) C:\InterSystems\MyIRIS\mgr\irisaudit\
4) C:\InterSystems\MyIRIS\mgr\irislib\
5) C:\InterSystems\MyIRIS\mgr\iristemp\
6) C:\InterSystems\MyIRIS\mgr\encrypted1\
7) C:\InterSystems\MyIRIS\mgr\encrypted2\
8) C:\InterSystems\MyIRIS\mgr\unencrypted\

Source Directory for Copy (? for List)?
```

暗号化データベースの数 (7 など) を入力します。

e. ^GBLOCKCOPY でデータのコピー先ディレクトリの入力を求められたら、暗号化されていないデータベースの名前を入力するか、? と入力してソース・ディレクトリのリストに類似したリストを表示します。

f. ^GBLOCKCOPY ですべてのグローバルをコピーするかどうかを尋ねられたら、Yes (Yes、Y、y などとすることができます) と入力します。

```
All Globals? No => y
```

g. 空のグローバルがデータベースにある場合、^GBLOCKCOPY で、これをコピーするかどうかを尋ねられます。これは、以下のように表示されます。

```
All Globals? No => y

^oddBIND     contains no data
Include it anyway? No =>
```

既定値の No (No、N、n などとすることができます) を入力します。

h. ^GBLOCKCOPY で、他の空のグローバルをすべてスキップするかどうかを尋ねられます。既定値の Yes (Yes、Y、y などとすることができます) を入力します。

```
Exclude any other similar globals without asking again? Yes =>
```

コピーされていないすべての空のグローバルのリストが表示されます。

```
Exclude any other similar globals without asking again? Yes => Yes
^oddCOM      contains no data --  not included
^oddDEP      contains no data --  not included
^oddEXT      contains no data --  not included
^oddEXTR     contains no data --  not included
^oddMAP      contains no data --  not included
^oddPKG      contains no data --  not included
^oddPROC     contains no data --  not included
```

```
^oddPROJECT  contains no data --  not included
^oddSQL      contains no data --  not included
^oddStudioDocument contains no data --  not included
^oddStudioMenu contains no data --  not included
^oddTSQL     contains no data --  not included
^oddXML      contains no data --  not included
^rBACKUP     contains no data --  not included
^rINC        contains no data --  not included
^rINCSAVE    contains no data --  not included
^rINDEXEXT   contains no data --  not included
^rINDEXSQL   contains no data --  not included
^rMACSAVE    contains no data --  not included
9 items selected from
29 available globals
```

i. ^GBLOCKCOPY で、この処理でジャーナリングを無効にするかどうかを尋ねられます。

```
Turn journaling off for this copy? Yes =>
```

既定値の Yes (Yes、Y、y などとすることができます) を入力します。

j. ^GBLOCKCOPY で、データをコピーすることの確認を求められます。

```
Confirm copy? Yes =>
```

既定値の Yes (Yes、Y、y などとすることができます) を入力します。データベースのサイズとプロセッサの処理速度によっては、コピーのステータスが進捗状況に応じて表示される場合があります。完了すると、^GBLOCKCOPY で以下のようなメッセージが表示されます。

```
Copy of data has completed
```

k. ^GBLOCKCOPY で、コピーに関連付けられた統計を保存するかどうかを尋ねられます。既定値の No (No、N、n などとすることができます) を入力します。

```
Do you want to save statistics for later review? No =>
```

コントロールはメイン・プロンプトに戻ります。

5. コピーされたデータが有効かどうかをテストします。このためには、管理ポータルのシステム・エクスプローラで、^GBLOCKCOPY におけるデータのコピー先のデータベースについて、クラス、テーブル、またはグローバルを調査します。
6. データが有効な場合、アクセス不能なキーまたは損傷したキーで暗号化されたデータベースごとに、この手順の 3 と 4 を実行します。
7. 暗号化されていないデータベースにすべての暗号化データベースをコピーしたら、それぞれのデータベースの 2 つ目のコピーを、できればそれぞれの最初のコピーがあるマシンとは別のマシンに作成します。
8. これで、すべての暗号化データベースをディスマウントし、有効なキー (紛失または損傷したキー・ファイルのキー) を無効にできます。この操作は、この時点でのみ可能です。InterSystems IRIS では、すべての暗号化データベースのキーを無効にするには、先にそれらのデータベースをディスマウントする必要があります。

以上で、暗号化されていない 1 つ以上のデータベースにデータがあり、有効なキーが存在しない状態になります。

以前に暗号化されたデータベースを再暗号化するには、以下の手順に従います。

1. "キーの作成" で説明されている手順に従い、新しいデータベース暗号化キーを作成します。
2. "暗号化データのアクセスにおける偶発的な損失からの保護" で説明されているように、キー・ファイルの新しいバックアップ・コピーを作成します。

   注意　"暗号化データのアクセスにおける偶発的な損失からの保護" の説明にある注意事項が守られていることを確認します。この手順に従わないと、データが永久に失われる場合があります。

3. 新しいキーを使用して、新しい暗号化データベースを 1 つ以上作成します。
4. 前述の手順でエクスポートしたデータを新しい暗号化データベースにインポートします。

これで、正常なキーとそのキーがあるキー・ファイルのバップアップが存在する暗号化データベースにデータが格納されました。

起動時に必要なデータベース暗号化キー・ファイルが存在しない場合

起動時にデータベース暗号化キー・ファイルを使用する必要がある状況では、システムはシングル・ユーザ・モードで起動します。該当する状況は以下のとおりです。

- InterSystems IRIS がインタラクティブに起動または無人起動するように構成されている場合
- 起動時にジャーナル・ファイルまたは IRISTEMP データベースおよび IRISLOCALDATA データベース (あるいはこれらすべて) を暗号化するように指定されている場合、または起動時に暗号化データベースが必要であると指定されている場合
- データベース暗号化キー・ファイルが存在しない場合

キー・ファイルを使用可能にできる場合

InterSystems IRIS の起動時に適切なキー・ファイルが存在しないことだけが理由で (例えば、キー・ファイルを保持するメディアが現在使用できない場合)、この状況が発生する可能性があります。

この状況を改善するには、InterSystems IRIS の実行をシングル・ユーザ・モードで開始した後、以下の手順を実行します。

1. InterSystems IRIS をシャットダウンします。例えば、InterSystems IRIS インスタンスを“MyIRIS”とすると、シャットダウンを実行するコマンドは以下のようになります。

   ```
   iris force MyIRIS
   ```

2. InterSystems IRIS がデータベース暗号化キー・ファイルを検索する場所がわかっている場合は、その場所にキー・ファイルを配置します(わからない場合は、次のセクションに示すように ^STURECOV を実行する必要があります)。
3. InterSystems IRIS を再起動します。

InterSystems IRIS は通常モード (マルチ・ユーザ・モード) で起動し、適切に動作します。

バックアップ・キー・ファイルが使用可能な場合

InterSystems IRIS の起動時に適切なキー・ファイルが存在しないために使用できない場合、バックアップ・キー・ファイルを使用できます。この場合、状況を改善するには、InterSystems IRIS の実行をシングル・ユーザ・モードで開始した後、以下の手順を実行します。

1. インターシステムズのサポート窓口までお問い合わせください。サポート窓口のエンジニアが以下の手順をご案内して、あらゆるご質問にお答えします。
2. messages.log ファイルの最新エントリにある手順に従い、管理者ターミナル・セッションを開始します。通常、-B フラグでターミナル・セッションを開始するように指定されています。

   例えば、“MyIRIS”という InterSystems IRIS インスタンスが既定の場所にインストールされている場合、Windows のコマンド行で次のコマンドを入力します。

   ```
   c:\InterSystems\MyIRIS\bin\irisdb -sc:\InterSystems\MyIRIS\mgr -B
   ```

   オペレーティング・システムのターミナル・ウィンドウで InterSystems IRIS に接続します。このウィンドウのプロンプトは、オペレーティング・システムのプロンプトから InterSystems IRIS の %SYS プロンプトに変更されます。

3. データベース暗号化キー・ファイルのコピー (バックアップなど) があるか、入手できる場合、InterSystems IRIS にアクセス可能な場所にそのキー・ファイルのコピーを置きます。
4. ターミナル・プロンプトで、^STURECOV (スタートアップ・リカバリ) ルーチンを実行します。このルーチンでは、そのファイルの管理者のユーザ名とパスワードを使用して暗号化キーを有効にします(このプロセスが完了したときに、^STURECOV を終了する必要はありません)。
5. InterSystems IRIS を使用する準備ができたら、^STURECOV を使用してスタートアップ・プロシージャを完了します。マルチ・ユーザ・モードで InterSystems IRIS を起動します。

以上で、InterSystems IRIS が適切に動作します。

キー・ファイルが使用できない場合

データベース暗号化キー・ファイルのコピーがない場合、以下の手順を実行します。

1. インターシステムズのサポート窓口までお問い合わせください。サポート窓口のエンジニアが以下の手順をご案内して、あらゆるご質問にお答えします。
2. messages.log ファイルの最新エントリにある手順に従い、管理者ターミナル・セッションを開始します。通常、-B フラグでターミナル・セッションを開始するように指定されています。

   例えば、“MyIRIS” という InterSystems IRIS インスタンスが既定の場所にインストールされている場合、Windows のコマンド行で次のコマンドを入力します。

   ```
   c:\InterSystems\MyIRIS\bin\irisdb -sc:\InterSystems\MyIRIS\mgr -B
   ```

   オペレーティング・システムのターミナル・ウィンドウで InterSystems IRIS に接続します。このウィンドウのプロンプトは、オペレーティング・システムのプロンプトから InterSystems IRIS の %SYS プロンプトに変更されます。
3. 起動時にマウントする必要がある暗号化データベースがある場合、それらに対してこの機能を無効にします。
   a. 管理ポータルのホーム・ページで、[**ローカルデータベース**] ページ ([**システム管理**] > [**構成**] > [**システム構成**] > [**ローカルデータベース**]) に移動します。
   b. データベースのテーブルでデータベース名をクリックします。そのデータベースの [**編集**] ページが表示されます。
   c. [**編集**] ページで、[**起動時にマウントが必要**] チェック・ボックスのチェックを外します。
   d. [**保存**] をクリックします。
4. ターミナル・プロンプトで、^STURECOV ルーチンを実行します。そのルーチンで、データベース暗号化キーを必要としないように InterSystems IRIS データベースの起動オプションを構成します。これは、**IRISTEMP** データベースおよび **IRISLOCALDATA** データベースとジャーナル・ファイルが適切に動作して、暗号化データベースをマウントできないことを表します。
5. InterSystems IRIS を使用する準備ができたら、^STURECOV を使用してスタートアップ・プロシージャを完了します。マルチ・ユーザ・モードで InterSystems IRIS を起動します。

この手順を実行する際、サポート窓口の担当者の指示に従い、その他の操作を実行する必要がある場合があります。担当者の指示に従ってください。

注意　”暗号化データのアクセスにおける偶発的な損失からの保護” で説明されている操作を実行しないと、いずれの形式でもデータを使用できなくなる可能性があります。これは非常に深刻な問題ですが、キーがないと、失われたデータを取得する方法がありません。

KMIP サーバ使用時における緊急事態への対処

ここでは、KMIP サーバの使用時にデータを損失する恐れがある事態において、どのように対処するべきかについて説明します。これらの状況には以下が含まれます。

- 有効なキーが格納されている KMIP サーバが損傷または紛失した場合
  - KMIP サーバ上のキーのバックアップ・コピーがある場合
  - KMIP サーバ上のキーのバックアップ・コピーがない場合

    警告　これは緊急事態です。即座に対処してください。

- 起動時に KMIP サーバが必要で、KMIP サーバにアクセスできない場合
  - KMIP サーバへの接続が一時的に利用不可能である場合
  - KMIP サーバで長期的な障害が発生している場合

有効なキーが格納されている KMIP サーバが損傷または紛失した場合

この場合、以下の状況が発生します。

- データベース暗号化キーが InterSystems IRIS インスタンスに対して有効化されています。
- InterSystems IRIS が暗号化データを使用します。
- データベース暗号化キーが格納されている KMIP サーバが破損します。

KMIP サーバ上のキーのバックアップ・コピーがある場合

有効なキーが格納されている KMIP サーバがアクセス不能になったり損傷したりした場合、直ちにベンダの指示に従って KMIP サーバのリストア手順を実行します。

KMIP サーバ上のキーのバックアップ・コピーがない場合

警告　この手順は、InterSystems IRIS データベースの暗号化データが失われる恐れがある緊急事態に実行します。

有効なキーが格納されている KMIP サーバをバックアップからリストアする方法がない場合、そのキーで暗号化されている各データベースに対し、直ちに以下の手順を実行します。

1. 警告　InterSystems IRIS をシャットダウンしたり有効になっているキーを無効にしたりすると、データが永久に失われます。

   InterSystems IRIS をシャットダウンしないでください。

   現在有効になっているキーは無効にしないでください。

2. インターシステムズのサポート窓口までお問い合わせください。サポート窓口のエンジニアが以下の手順をご案内して、あらゆるご質問にお答えします。
3. データベースをディスマウントします。これにより、暗号化されていないデータベースにデータをコピーしている最中に、暗号化されたコンテンツがあるデータベースではどのユーザも変更ができないようにします。
   a. 管理ポータルのホーム・ページで、**[データベース]** ページ (**[システム処理]** > **[データベース]**) に移動します。
   b. 暗号化データベースがマウントされている場合、**[データベース]** ページで、データベースの行にある最後から 2 番目の列で **[ディスマウント]** オプションを選択します。確認ダイアログで [OK] をクリックします。
   c. **[データベース]** ページが再び表示されたら、データベースの行にある最後の列で **[マウント]** オプションを選択します。

d. [**データベースマウント**] 確認画面で、[**読み取り専用**] チェック・ボックスにチェックを付け、[OK] を選択します。

この手順の間、誰もデータベースに対して変更を行わないことが重要です。データベースを読み取り専用でマウントすると、ユーザはデータを変更できなくなります。

4. 暗号化されていない状態ですべてのデータを別のデータベースにコピーします。データをコピーする手順は以下のとおりです。

a. ターミナルで、%SYS ネームスペースに移動します。

```
REGULARNAMESPACE>set $namespace="%SYS"
```

b. そのネームスペースで ^GBLOCKCOPY コマンドを実行します。

```
%SYS>do ^GBLOCKCOPY

This routine will do a fast global copy from a database to another database or
to a namespace. If a namespace is the destination, the global will follow any
mappings set up for the namespace.

1) Interactive copy
2) Batch copy
3) Exit

Option?1
```

c. ^GBLOCKCOPY プロンプトで、インタラクティブ・コピーとして 1 を指定します。

```
Option? 1

1) Copy from Database to Database
2) Copy from Database to Namespace
3) Exit

Option?
```

d. ^GBLOCKCOPY プロンプトで、コピー・タイプの入力が要求されたら、別のデータベースへのコピーを指定する 1 を選択します。

```
Option? 1
Source Directory for Copy (? for List)?
```

ここで、暗号化データベースの名前を指定するか、? と入力して番号付けされたデータベースのリストを表示します。リストには暗号化データベースが含まれます。? と入力すると、^GBLOCKCOPY は以下のようなリストを表示します。

```
Source Directory for Copy (? for List)? ?

1) C:\InterSystems\MyIRIS\mgr\
2) C:\InterSystems\MyIRIS\mgr\irislocaldata\
3) C:\InterSystems\MyIRIS\mgr\irisaudit\
4) C:\InterSystems\MyIRIS\mgr\irislib\
5) C:\InterSystems\MyIRIS\mgr\iristemp\
6) C:\InterSystems\MyIRIS\mgr\encrypted1\
7) C:\InterSystems\MyIRIS\mgr\encrypted2\
8) C:\InterSystems\MyIRIS\mgr\unencrypted\

Source Directory for Copy (? for List)?
```

暗号化データベースの数 (7 など) を入力します。

e. ^GBLOCKCOPY でデータのコピー先ディレクトリの入力を求められたら、暗号化されていないデータベースの名前を入力するか、? と入力してソース・ディレクトリのリストに類似したリストを表示します。

f. ^GBLOCKCOPY ですべてのグローバルをコピーするかどうかを尋ねられたら、Yes (Yes、Y、y などとすることができます) と入力します。

```
All Globals? No => y
```

g. 空のグローバルがデータベースにある場合、^GBLOCKCOPY で、これをコピーするかどうかを尋ねられます。これは、以下のように表示されます。

```
All Globals? No => y

^oddBIND     contains no data
Include it anyway? No =>
```

既定値の No (No、N、n などとすることができます) を入力します。

h. ^GBLOCKCOPY で、他の空のグローバルをすべてスキップするかどうかを尋ねられます。既定値の Yes (Yes、Y、y などとすることができます) を入力します。

```
Exclude any other similar globals without asking again? Yes =>
```

コピーされていないすべての空のグローバルのリストが表示されます。

```
Exclude any other similar globals without asking again? Yes => Yes
^oddCOM      contains no data --  not included
^oddDEP      contains no data --  not included
^oddEXT      contains no data --  not included
^oddEXTR     contains no data --  not included
^oddMAP      contains no data --  not included
^oddPKG      contains no data --  not included
^oddPROC     contains no data --  not included
^oddPROJECT  contains no data --  not included
^oddSQL      contains no data --  not included
^oddStudioDocument contains no data --  not included
^oddStudioMenu contains no data --  not included
^oddTSQL     contains no data --  not included
^oddXML      contains no data --  not included
^rBACKUP     contains no data --  not included
^rINC        contains no data --  not included
^rINCSAVE    contains no data --  not included
^rINDEXEXT   contains no data --  not included
^rINDEXSQL   contains no data --  not included
^rMACSAVE    contains no data --  not included
9 items selected from
29 available globals
```

i. ^GBLOCKCOPY で、この処理でジャーナリングを無効にするかどうかを尋ねられます。

```
Turn journaling off for this copy? Yes =>
```

既定値の Yes (Yes、Y、y などとすることができます) を入力します。

j. ^GBLOCKCOPY で、データをコピーすることの確認を求められます。

```
Confirm copy? Yes =>
```

既定値の Yes (Yes、Y、y などとすることができます) を入力します。データベースのサイズとプロセッサの処理速度によっては、コピーのステータスが進捗状況に応じて表示される場合があります。完了すると、^GBLOCKCOPY で以下のようなメッセージが表示されます。

```
Copy of data has completed
```

k. ^GBLOCKCOPY で、コピーに関連付けられた統計を保存するかどうかを尋ねられます。既定値の No (No、N、n などとすることができます) を入力します。

```
Do you want to save statistics for later review? No =>
```

コントロールはメイン・プロンプトに戻ります。

5. コピーされたデータが有効かどうかをテストします。このためには、管理ポータルのシステム・エクスプローラで、^GBLOCKCOPY におけるデータのコピー先のデータベースについて、クラス、テーブル、またはグローバルを調査します。

6. データが有効な場合、アクセス不能なキーまたは損傷したキーで暗号化されたデータベースごとに、この手順の 3 と 4 を実行します。
7. 暗号化されていないデータベースにすべての暗号化データベースをコピーしたら、それぞれのデータベースの 2 つ目のコピーを、できればそれぞれの最初のコピーがあるマシンとは別のマシンに作成します。
8. これで、すべての暗号化データベースをディスマウントし、有効なキー (紛失または損傷したキー・ファイルのキー) を無効にできます。この操作は、この時点でのみ可能です。InterSystems IRIS では、すべての暗号化データベースのキーを無効にするには、先にそれらのデータベースをディスマウントする必要があります。

以上で、暗号化されていない 1 つ以上のデータベースにデータがあり、有効なキーが存在しない状態になります。

以前に暗号化されたデータベースを再暗号化するには、以下の手順に従います。

1. "KMIP サーバ上のキーの作成" で説明されている手順に従い、新しいデータベース暗号化キーを作成します。
2. "暗号化データのアクセスにおける偶発的な損失からの保護" で説明されているように、キー・ファイルの新しいバックアップ・コピーを作成します。

   注意　"暗号化データのアクセスにおける偶発的な損失からの保護" の説明にある注意事項が守られていることを確認します。この手順に従わないと、データが永久に失われる場合があります。

3. 新しいキーを使用して、新しい暗号化データベースを 1 つ以上作成します。
4. 前述の手順でエクスポートしたデータを新しい暗号化データベースにインポートします。

これで、正常なキーとそのキーがあるキー・ファイルのバップアップが存在する暗号化データベースにデータが格納されました。

起動時に KMIP サーバが必要で、KMIP サーバにアクセスできない場合

起動時に 1 つ以上のデータベース暗号化キーを使用する必要がある状況では、システムはシングル・ユーザ・モードで起動します。該当する状況は以下のとおりです。

- InterSystems IRIS がインタラクティブに起動または無人起動するように構成されている場合
- 起動時にジャーナル・ファイルまたは IRISTEMP データベースおよび IRISLOCALDATA データベース (あるいはこれらすべて) を暗号化するように指定されている場合、または起動時に暗号化データベースが必要であると指定されている場合
- 必要なデータベース暗号化キーが格納されている KMIP サーバがアクセス不能である場合

KMIP サーバへの接続が一時的に利用不可能である場合

ネットワーク障害などの問題が発生している場合や、他の理由で KMIP サーバが一時的に動作していない場合、最も簡単な解決策は、ネットワークやサーバの問題を解決し、必要に応じて InterSystems IRIS を再起動することです。

KMIP サーバで長期的な障害が発生している場合

KMIP サーバに長時間接続できない場合は、以下の手順に従います。

1. messages.log ファイルの最新エントリにある手順に従い、管理者ターミナル・セッションを開始します。通常、-B フラグでターミナル・セッションを開始するように指定されています。

   例えば、“MyIRIS” という InterSystems IRIS インスタンスが既定の場所にインストールされている場合、Windows のコマンド行で次のコマンドを入力します。

   ```
   c:\InterSystems\MyIRIS\bin\irisdb -sc:\InterSystems\MyIRIS\mgr -B
   ```

   オペレーティング・システムのターミナル・ウィンドウで InterSystems IRIS に接続します。このウィンドウのプロンプトは、オペレーティング・システムのプロンプトから InterSystems IRIS の %SYS プロンプトに変更されます。

2. 起動時にマウントする必要がある暗号化データベースがある場合、それらに対してこの機能を無効にします。

   a. 管理ポータルのホーム・ページで、[**ローカルデータベース**] ページ ([**システム管理**] > [**構成**] > [**システム構成**] > [**ローカルデータベース**]) に移動します。

   b. データベースのテーブルでデータベース名をクリックします。そのデータベースの [**編集**] ページが表示されます。

   c. [**編集**] ページで、[**起動時にマウントが必要**] チェック・ボックスのチェックを外します。

   d. [**保存**] をクリックします。

3. ターミナル・プロンプトで、^STURECOV ルーチンを実行します。そのルーチンで、データベース暗号化キーを必要としないように InterSystems IRIS データベースの起動オプションを構成します。これは、**IRISTEMP** データベースおよび **IRISLOCALDATA** データベースとジャーナル・ファイルが適切に動作して、暗号化データベースをマウントできないことを表します。

4. InterSystems IRIS を使用する準備ができたら、^STURECOV を使用してスタートアップ・プロシージャを完了します。マルチ・ユーザ・モードで InterSystems IRIS を起動します。

注意　　"暗号化データのアクセスにおける偶発的な損失からの保護" で説明されている操作を実行しないと、いずれの形式でもデータを使用できなくなる可能性があります。これは非常に深刻な問題ですが、キーがないと、失われたデータを取得する方法がありません。

暗号化に関するその他の情報

暗号化に関するその他の情報

ここでは、InterSystems IRIS® データ・プラットフォームでの暗号化に関するその他の情報を紹介します。

キー・ファイル暗号化情報

データベース暗号化の管理者名は、キー・ファイルに平文で保存されています。データベース暗号化の管理者パスワードは保存されていません。パスワードを入力すると、他のデータと共にそれを使用してキー暗号化キーが生成されます。有効なパスワードを第三者が推測できる場合、そのパスワードのポリシーは脆弱すぎます。キー暗号化キーは、512 ビットの salt と 65,536 回の反復処理を使用する PBKDF2 アルゴリズムを使用して生成され、ディクショナリおよび総当り攻撃を実行不可能にします。

暗号化とデータベース関連機能

InterSystems IRIS のデータベース暗号化では、データベースのファイルそのものが保護されます。InterSystems IRIS の関連機能は以下のとおりです。

- InterSystems IRIS のオンライン・バックアップは暗号化されません。InterSystems IRIS データベースのバックアップを暗号化するには、InterSystems IRIS を停止し、ファイル・システムのバックアップを実行することをお勧めします ("外部バックアップ" を参照)。
- 暗号化データベースのブロックは、ライト・イメージ・ジャーナル (WIJ) ファイルに暗号化されます。
- **IRISTEMP** データベースおよび **IRISLOCALDATA** データベースは、オプションで暗号化できます。**IRISTEMP** および **IRISLOCALDATA** を暗号化する方法は、"暗号化の起動設定の構成" を参照してください。
- オプションでジャーナル・ファイルを暗号化できます。"暗号化の起動設定の構成" を参照してください。

暗号化、ハッシュ、およびその他のキー関連操作を実行するための呼び出しについて

InterSystems IRIS では、**%SYSTEM.Encryption** クラスのさまざまなメソッドを使用して、データ暗号化、Base64 エンコーディング、ハッシュ、およびメッセージ認証コードの生成に関連するアクションを実行できます。これには、AES 暗号化、各種の RSA アルゴリズム、SHA-256 ハッシュ関数などを呼び出すメソッドが含まれます。いくつかの呼び出しを以下に示します。

- $System.Encryption.AESCBCManagedKeyEncrypt および $System.Encryption.AESCBCManagedKeyDecrypt
- $System.Encryption.AESKeyWrap および $System.Encryption.AESKeyUnwrap
- $System.Encryption.Base64Encode および $System.Encryption.Base64Decode
- $System.Encryption.RSASHASign および $System.Encryption.RSASHAVerify
- $System.Encryption.RSAEncrypt および $System.Encryption.RSADecrypt
- $System.Encryption.SHAHash

RSAEncrypt および RSADecrypt の使用例

以下に、RSAEncrypt および RSADecrypt の呼び出しの使用例を示します。これは以下を前提としています。

- コードは Windows で実行されます。
- 使用可能な証明書、秘密鍵、および認証機関 (CA) 証明書があります(この例を試すには、これらを入手する必要があります)。
- これら 3 つのアイテムはすべて **C:¥Keys¥** ディレクトリにあります。

処理の詳細は、例の中のコメントを参照してください。

ObjectScript

```
set dir = "C:\Keys\"

// certificate for the instance performing encryption and decryption
// and private key associated with that above certificate
set cert = dir_"test.crt"
set key = dir_"test.key"

// certificate for the CA of the instance
set cacert=dir_"ca.crt"

set data = "data to be encrypted"

// create a local set of X.509 credentials with the
// certificate and private key
set credentials = ##class(%SYS.X509Credentials).%New()
set credentials.Alias="TestCreds"
write credentials.LoadCertificate(cert)
write credentials.LoadPrivateKey(key)
write credentials.Save(),!

// encrypt the data using the public key in the certificate, write it
// to the display, and display error information, if there is any
set ciphertext=$System.Encryption.RSAEncrypt(data,credentials.Certificate,cacert)
write ciphertext,!
write $System.Encryption.RSASHA1GetLastError()

// decrypt the data using the private key, write it to the display,
// and display error information, if there is any
write "now decrypting -=-=-=-=-=-=-=-=-=-=-",!
set cleartext=$System.Encryption.RSADecrypt(ciphertext,credentials.PrivateKey)
write cleartext,!
write $System.Encryption.RSASHA1GetLastError()
```

FIPS-2 準拠

データベース暗号化の FIPS 140-2 準拠

特定のプラットフォームでは、InterSystems IRIS® データ・プラットフォームは FIPS 140-2 に準拠したデータベース暗号化をサポートします(FIPS 140-2 は Federal Information Processing Standard Publication 140-2 を指します。詳細は、http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf で確認できます)。

このバージョンの InterSystems IRIS は、x86-64 対応の Red Hat Enterprise Linux 8 で、FIPS 140-2 に準拠したデータベース暗号化をサポートしています。Red Hat は、OpenSSL **libcrypto.so** ライブラリおよび **libssl.so** ライブラリの検証証明書を保有しています。FIPS モードでの実行時には、InterSystems IRIS はこれらの証明書ライブラリを使用します。Red Hat Linux のマイナー・バージョンに最新の証明書があるかどうかを確認するには、Red Hat のドキュメントを参照してください。

注釈　FIPS モードが有効な場合の動作:

- Red Hat 8 では TLSv1.2 と TLSv1.3 のみがサポートされます。

政府規格に対する Red Hat のサポートについては、https://access.redhat.com/articles/2918071 を参照してください。

重要　InterSystems IRIS では、現在 Red Hat 9 での FIPS モードはサポートしていません。

FIPS サポートの有効化

InterSystems IRIS で、FIPS 140-2 に準拠したデータベース暗号化のサポートを有効にするには、以下を実行します。

1. Red Hat リポジトリから **openssl** パッケージ (rhel-8-server-rpms) をダウンロードしてインストールします。
2. オペレーティング・システムの FIPS モードを有効にします。手順については、Red Hat の Web サイトの記事 "RHEL 6/7/8 を FIPS 140-2 準拠に設定するにはどうすればよいですか?" を参照してください(この記事にアクセスするには、Red Hat のログイン資格情報が必要です)。
3. ディレクトリ **/usr/lib64** で、以下のシンボリック・リンクを確認します。シンボリック・リンクが存在しない場合、作成します。
   - シンボリック・リンク **libssl.so.1.1** は、同じディレクトリ内の適切なファイル (**libssl.so.1.1.1g** ファイルなど) を指している必要があります。
   - シンボリック・リンク **libcrypto.so.1.1** は、同じディレクトリ内の適切なファイル (**libcrypto.so.1.1.1g** ファイルなど) を指している必要があります。
4. InterSystems IRIS で、**FIPSMode** CPF パラメータに **True** (1) を指定します。そのためには、以下の操作を実行します。
   a. 管理ポータルを開きます。
   b. **[システム管理]→[構成]→[追加設定]→[開始]** を選択します。

      FIPSMode の行が表示されます。
   c. **FIPSMode** の値を **True** に指定して、変更内容を保存します。
5. InterSystems IRIS を再起動します。
6. "暗号化データベースの使用法" で説明されているように、暗号化データベースを有効化して構成します。

開始動作と messages.log

InterSystems IRIS の開始時 :

- FIPSMode が 0 の場合、InterSystems IRIS ネイティブの暗号化が使用されます。Intel AES-NI ハードウェア命令を使用する、最適化されたアセンブリー・コードなどです (CPU がサポートする場合)。このモードでは、InterSystems IRIS は開始時、messages.log に以下を書き込みます。

  Terminal

  ```
  FIPS 140-2 compliant cryptography for database encryption is not configured in iris.cpf
  ```

- FIPSMode が 1 の場合、InterSystems IRIS は /usr/lib64/libcrypto.so FIPS 検証済みライブラリ内の関数への参照を解決しようとします。その後、FIPS モードでライブラリを初期化しようとします。これらの手順が成功すると、InterSystems IRIS は messages.log に以下を書き込みます。

  Terminal

  ```
  FIPS 140-2 compliant cryptography for database encryption is enabled for this instance.
  ```

- FIPSMode が 1 で、ライブラリの初期化が失敗した場合、InterSystems IRIS は開始されません。この場合、messages.log には以下のメッセージが書き込まれます。

  Terminal

  ```
  FIPS 140-2 compliant cryptography for database encryption initialization failed. Aborting.
  ```

- lnxrhx64 以外のプラットフォームでは、FIPSMode が 1 の場合、InterSystems IRIS ネイティブの暗号化が使用され、InterSystems IRIS はmessages.log に以下を書き込みます。

  Terminal

  ```
  FIPS 140-2 compliant cryptography for database encryption is not supported on this platform.
  ```

暗号化標準および RFC

暗号化標準および RFC

インターシステムズのセキュリティで使用されている暗号化の基本とアルゴリズムは、以下の基準と RFC (Requests for Comments) で定義されています。

- AES (Advanced Encryption Standard) 暗号化 ー FIPS (Federal Information Processing Standards) 197
- AES Key Wrap ー
  - NIST (National Institute of Standards and Technology) のドキュメント“Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping”(https://csrc.nist.gov/CryptoToolkit/kms/AES_key_wrap.pdf)
  - IETF (Internet Engineering Task Force) RFC 3394
- Base64 エンコード ー RFC 3548
- ブロック埋め込み ー PKCS (Public-Key Cryptography Standard) #7 および RFC 2040
- CBC (Cipher Block Chaining) 暗号化モード ー NIST 800-38A
- 決定論的乱数ジェネレータ ー
  - FIPS PUB 140-2 の Annex C
  - FIPS PUB 186-2 の Change Notice 1、Appendix 3.1、および Appendix 3.3
- GSS (Generic Security Services) API ー
  - The Kerberos Version 5 GSS-API Mechanism ー RFC 1964
  - Generic Security Service Application Program Interface, Version 2, Update 1 ー RFC 2743
  - Generic Security Service API Version 2: C Bindings ー RFC 2744
  - Generic Security Service API Version 2: Java Bindings ー RFC 2853
- Kerberos Network Authentication Service (V5) ー RFC 1510
- Hash-based Message Authentication Code (HMAC) ー FIPS 198 および RFC 2104
- Message Digest 5 (MD5) hash ー RFC 1321
- Password-Based Key Derivation Function 2 (PBKDF2) ー PKCS #5 v2.1 および RFC 8018
- Secure Hash Algorithm (SHA-1) ー FIPS 180-2 および RFC 3174
- Secure Hash Algorithm (SHA-512) ー FIPS 180-2 および RFC 6234

これらのドキュメントはすべてオンラインで入手できます。

- FIPS ドキュメント
- NIST ドキュメント
- RFC (IETF)

公開鍵インフラストラクチャ

インターシステムズ公開鍵インフラストラクチャ

重要　　インターシステムズの PKI はテストのみを目的としています。運用設定では使用しないでください。

このドキュメントでは、以下の項目について説明します。

- インターシステムズ公開鍵インフラストラクチャ (PKI) について
- 認証機関サーバ・タスク
  - 認証機関サーバとしての InterSystems IRIS® インスタンスの構成
  - 保留中の証明書署名要求の管理
- 認証機関クライアント・タスク
  - 認証機関クライアントとしての InterSystems IRIS インスタンスの構成
  - 認証機関サーバへの証明書署名要求の送信
  - 認証機関サーバからの証明書の取得

インターシステムズ公開鍵インフラストラクチャ (PKI) について

公開鍵インフラストラクチャ (PKI) は、秘密鍵、公開鍵、および証明書を作成、管理する手段を提供します。これらは、暗号化、解読、デジタル署名、シグニチャの検証などの暗号処理に使用されます。証明書は、公開鍵を識別情報と関連付ける手段を提供します。このために、PKI には認証局 (CA) と呼ばれる信頼されるサード・パーティが含まれます。

インターシステムズの PKI 実装では、InterSystems IRIS が認証機関 (CA) として機能できます。CA として機能する InterSystems IRIS のインスタンスは CA サーバと呼ばれ、CA のサービスを利用する InterSystems IRIS のインスタンスは CA クライアントと呼ばれます。InterSystems IRIS の 1 つのインスタンスは、CA サーバにも CA クライアントにもなることができます。CA サーバとしてのインスタンスは、自己署名 CA 証明書を生成して使用するか、商用のサード・パーティまたは製品により発行された CA 証明書を使用できます。 CA クライアントとしてのインスタンスは、CA サーバと関連付けられます。CA クライアントの証明書は、TLS、XML 暗号化、およびシグニチャの検証で使用できます。中間 CA として機能するように CA クライアントを構成することもできます。PKI が関与する通信は、Web サービスで発生します。

自身を CA サーバとして確立すると、InterSystems IRIS のインスタンスは、公開鍵と秘密鍵のペアを作成してその公開鍵を自己署名 X.509 証明書に埋め込むか、外部 CA により署名された X.509 証明書と秘密鍵を使用します。X.509 は業界標準の証明書構造で、公開鍵をエンティティとその他関連データの両方の識別情報に関連付けます。この識別情報は所有者識別名 (DN) と呼ばれ、エンティティの組織、場所、またはその両方に関するさまざまな固有情報で構成されます。X.509 証明書を使用して公開鍵ベースの高レベルのセキュリティを提供できるのは、秘密鍵の保護および証明書の発行に関する適切なセキュリティ・ポリシーが適用されている場合のみです。これには、CA サーバの秘密鍵ファイルの厳密な制御も含まれ、使用していないときには物理的に保護されるリムーバブル・メディアに保存することが推奨されます。

管理ポータルから InterSystems IRIS PKI インフラストラクチャを使用する場合、すべてのアクションは **[公開鍵インフラストラクチャ]** ページ (**[システム管理]** > **[セキュリティ]** > **[公開鍵インフラストラクチャ]**) から開始します。

PKI および CA の詳細は、“公開鍵インフラストラクチャ (PKI) について” を参照してください。インターシステムズの PKI の基礎となる TLS 呼び出しの技術的な詳細は、″OpenSSL″ ライブラリを参照してください。X.509 証明書の技術的な詳細は、″RFC 5280″ の “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile” を参照してください。

管理ポータルの PKI タスクのヘルプ

PKI タスクのヘルプへのリンクを以下に示します。

- 認証機関クライアント
  - 認証機関サーバへの証明書署名要求の送信
  - 認証機関サーバからの証明書の取得
  - ローカル認証機関クライアントの構成
- 認証機関サーバ
  - 保留中の証明書署名要求の処理
  - ローカル認証機関サーバの構成

認証機関サーバ・タスク

InterSystems IRIS インスタンスは、認証機関 (CA) サーバとして機能することができます。これには以下が含まれます。

1. CA サーバとしての InterSystems IRIS インスタンスの構成。これには、証明書に関する情報または、CA サーバが証明書署名要求の処理に使用する情報の提供が必要となります。
2. 保留中の証明書署名要求 (CSR) の管理。これは CA サーバの継続的な作業です。

注釈　これらは CA サーバ管理者のタスクであるため、このセクションはこれらの管理者を対象とします。CA クライアント・タスクはこれとは異なり、CA クライアントの管理者および技術担当者が担当します。

認証局サーバとしての InterSystems IRIS インスタンスの構成

PKI 処理を行うには、まず認証局 (CA) サーバとして InterSystems IRIS インスタンスを構成する必要があります。これには以下のいずれかが必要です。

- 新しい秘密鍵と証明書での CA サーバの構成
- 既存の秘密鍵と証明書での CA サーバの構成

CA サーバの再初期化が必要な場合もあります。

新しい秘密鍵と証明書での CA サーバの構成

新しい秘密鍵と証明書を作成している場合、手順は次のとおりです。

1. 選択した InterSystems IRIS インスタンスで、管理ポータルの **[公開鍵インフラストラクチャ]** ページ (**[システム管理]** > **[セキュリティ]** > **[公開鍵インフラストラクチャ]**) に移動します。
2. **[公開鍵インフラストラクチャ]** ページの **[認証機関サーバ]** で、**[ローカル認証機関サーバの構成]** を選択します。これにより、(1) CA サーバの証明書および秘密鍵のファイル名ルート、(2) これらのファイルがあるディレクトリの各フィールドが表示されます。

   重要　既存の証明書と秘密鍵を指すパスおよびファイル名ルートを指定する場合、InterSystems IRIS はその CA サーバの証明書と秘密鍵を使用します。それ以外の場合、InterSystems IRIS は新しい証明書と秘密鍵を作成します(また、いずれか 1 つのファイルのみが存在する場合、InterSystems IRIS はこれに `.old` 接尾辞を付加して名前を変更し、新しいファイルを作成します)。

   フィールドは以下のとおりです。

   - **[認証機関の証明書と秘密鍵ファイルのファイル名ルート (拡張子なし)]** — 必須。秘密鍵と証明書のファイル名の共通部分を指定します。これは、既存のファイルのペアのものでも新しいファイルのペアのものでもかまいません。したがって、ファイル名ルートとして **MyCA** を選択すると、秘密鍵は **MyCA.key** ファイルに保存され、証明書は **MyCA.cer** ファイルに保存されます。このフィールドで有効な文字は、英数字、ハイフン、およびアンダースコアです。ルートを文字列“cache”にすることはできません。

- **[認証機関の証明書と秘密鍵ファイルのディレクトリ]** ― 必須。CA の証明書と秘密鍵のファイルを格納するディレクトリへのパスです。ディレクトリが存在しない場合は、InterSystems IRIS がその作成を試みます。このディレクトリは、(ローカル・ハード・ドライブやネットワーク・サーバではなく) 外部デバイス (暗号化された外部デバイスを推奨) に存在する必要があります。これは CA の秘密鍵を保持するディレクトリであるため、完全に安全な場所であることが極めて重要です。ここで相対パスを指定すると、InterSystems IRIS インスタンスの <install-dir>/mgr/ を基準とした相対パスになります。

3. **[次]** をクリックして次に進みます。
4. 次に表示されるフィールドは、新しい秘密鍵と証明書ペアを作成しているのか、既存の秘密鍵と証明書を使用しているのかによって異なります。新しい秘密鍵と証明書を作成している場合、InterSystems IRIS には次のフィールドが表示されます。

   - **[認証機関の秘密鍵ファイルのパスワード]** および **[パスワード確認]** ― 必須。CA の秘密鍵ファイルを暗号化および解読するためのパスワードです。
   - **[認証機関所有者識別名]** ― CA 証明書のベアラを示す識別名 (DN) を定義する 1 つ以上の名前と値のペアのセットです。1 つ以上の属性の値を指定する必要があります。属性は以下のとおりです。
     - **[国]** ― ISO 国コードを使用して国を識別する 2 文字のコードです。
     - **[州または県]** ― CA のある州または県の名前です。通常、CA 証明書ではこのフィールドは使用しません。
     - **[地域]** ― CA のある市町村の名前です。通常、CA 証明書ではこのフィールドは使用しません。
     - **[組織]** ― CA を管理している組織の名前です。規則により、この値は単に“InterSystems”や“InterSystems Corp”ではなく、“InterSystems Corporation”のように略さずに入力します。
     - **[組織単位]** ― その他の組織情報または CA に関する特記事項です。ここには、CA の部署、CA は社内のみで使用するという記述などを入力できます。
     - **[共通名]** ― “ドキュメント・テスト CA”など、CA のわかりやすい名前です。
   - **[認証機関の証明書の有効期間 (日数)]** ― 必須。CA 証明書自体の有効期間 (存続期間) です。
   - **[認証機関により発行された証明書の有効期間 (日数)]** ― 必須。CA がそのクライアントに対して発行する証明書の有効期間 (存続期間) です。
   - **[電子メールの構成]** ― CA およびそのタスクを管理するための電子メール・アカウントで必要な情報です。
     - **[SMTP サーバ]** ― 完全修飾ホスト名の形式の、CA メールを処理するSimple Mail Transfer Protocol (SMTP) サーバです (“MyMachine.MyDomain.com”など)。
     - **[SMTPユーザ名]** ― 指定した SMTP サーバが認証できるユーザ名です。このフィールドでは完全修飾ユーザ名とする必要はありません。
     - **[SMTP パスワード]** ― SMTP ユーザ名に関連付けられているパスワードです。
     - **[パスワード確認]** ― SMTP ユーザ名に関連付けられているパスワードの確認です。
     - **[認証機関サーバ管理者の電子メール・アドレス]** ― CA の証明書署名要求を受け取るユーザです。このフィールドは、“CAMgr@MyDomain.com”のように完全修飾ユーザ名とする必要があります。

5. 必要に応じてこれらのフィールドに入力し、**[保存]** をクリックします。InterSystems IRIS には、成功したことを示す次のようなメッセージが表示されます。

```
Certificate Authority server successfully configured.
Created new files: C:\pki\FileNameRoot.cer .key, and .srl.
Certificate Authority Certificate SHA-1 fingerprint:
E3:FB:30:09:53:90:9A:31:30:D3:F0:07:8F:64:65:CD:11:0A:1A:A2
Confirmation email sent to: CAserver-admin@intersystems.com
```

これは、秘密鍵、証明書、およびそれらに関連付けられた SRL (シリアル) ファイルが作成されたことを示しています (それ以外の場合はエラー・メッセージを表示します)。

ファイルが作成されたら、物理的に保護されるリムーバブル・メディアに秘密鍵を保存することを強くお勧めします。

警告　すべての秘密鍵を適切に保護することが重要です。特に CA の秘密鍵の保護は最も重要です。秘密鍵が漏洩されると、セキュリティの侵害、データの漏洩、財務損失、および法的な脆弱性が生じる可能性があります。

成功した場合、InterSystems IRIS では以下のようなアクションが実行されたことになります。

- 鍵のペアの作成。
- 指定のファイル名ルートで、指定の場所のファイルに秘密鍵を保存 (下記参照)。
- 公開鍵を含む自己署名 CA 証明書の作成。
- 指定のファイル名ルートで、指定の場所のファイルに証明書を保存 (下記参照)。
- 発行された証明書数のカウンタを作成し、証明書や秘密鍵と同じディレクトリにある SRL (シリアル) ファイルに保存(CA が新しい証明書を発行するたびに、InterSystems IRIS はこのカウンタに基づいて証明書に一意のシリアル番号を付与し、SRL ファイル内の値をインクリメントします)。

CA 秘密鍵と証明書を作成したら、証明書署名要求 (CSR) を処理できます。CA クライアントが CSR を作成すると、CA 管理者はこれについての電子メール通知を受け取ります。

既存の秘密鍵と証明書での CA サーバの構成

(別の InterSystems IRIS CA や、商用 CA などの外部 CA などからの) 既存の秘密鍵と証明書を使用している場合、手順は以下のとおりです。

1. 秘密鍵と証明書を作成または取得します。証明書は PEM 形式であるか、PEM 形式に変換できることが必要です。
2. 証明書と秘密鍵にまだ同一のファイル名ルートがない場合、証明書は filenameroot`.cer`、秘密鍵は filenameroot`.key` という名前に変更します。この filenameroot が使用するファイル名ルートです。
3. 両方のファイルを同じディレクトリに保存します。このディレクトリには、CA サーバとして構成している InterSystems IRIS インスタンスからアクセスできることを確認してください。このディレクトリは、(ローカル・ハード・ドライブやネットワーク・サーバではなく) 外部デバイス (暗号化された外部デバイスを推奨) に存在する必要があります。これは CA の秘密鍵を保持するディレクトリであるため、完全に安全な場所であることが極めて重要です。

   警告　すべての秘密鍵を適切に保護することが重要です。特に CA の秘密鍵の保護は最も重要です。秘密鍵が漏洩されると、セキュリティの侵害、データの漏洩、財務損失、および法的な脆弱性が生じる可能性があります。

4. 選択した InterSystems IRIS インスタンスで、管理ポータルの **[公開鍵インフラストラクチャ]** ページ (**[システム管理]** > **[セキュリティ]** > **[公開鍵インフラストラクチャ]**) に移動します。
5. **[公開鍵インフラストラクチャ]** ページの **[認証機関サーバ]** で、**[ローカル認証機関サーバの構成]** を選択します。このページのフィールドに、以下のとおり入力します。
   - **[認証機関の証明書と秘密鍵ファイルのファイル名ルート (拡張子なし)]** – 必須。秘密鍵と証明書のファイル名の共通部分です。この値には、ファイルが持つファイル名ルート、または手順 2 で選択したファイル名ルートを使用します。
   - **[認証機関の証明書と秘密鍵ファイルのディレクトリ]** – 必須。CA の証明書と秘密鍵のファイルを保持するディレクトリへのパスです。この値には、手順 3 で選択したディレクトリを使用します。ここで相対パスを指定すると、InterSystems IRIS インスタンスの **<install-dir>/mgr/** を基準とした相対パスになります。
6. **[次]** をクリックして次に進みます。

7. 次に表示されるフィールドは、新しい秘密鍵と証明書ペアを作成しているのか、既存の秘密鍵と証明書を使用しているのかによって異なります。既存の秘密鍵と証明書を使用している場合、InterSystems IRIS には次のフィールドが表示されます。
   - **[認証機関により発行された証明書の有効期間（日数）]** ― 必須。CA がそのクライアントに対して発行する証明書の有効期間（存続期間）です。
   - **[電子メールの構成]** ― CA およびそのタスクを管理するための電子メール・アカウントで必要な情報です。
     - **[SMTP サーバ]** ― 完全修飾ホスト名の形式の、CA メールを処理するSimple Mail Transfer Protocol（SMTP）サーバです（“MyMachine.MyDomain.com” など）。
     - **[SMTPユーザ名]** ― 指定した SMTP サーバが認証できるユーザ名です。このフィールドでは完全修飾ユーザ名とする必要はありません。
     - **[SMTP パスワード]** ― SMTP ユーザ名に関連付けられているパスワードです。
     - **[パスワード確認]** ― SMTP ユーザ名に関連付けられているパスワードの確認です。
     - **[認証機関サーバ管理者の電子メール・アドレス]** ― CA の証明書署名要求を受け取るユーザです。このフィールドは、“CAMgr@MyDomain.com” のように完全修飾ユーザ名とする必要があります。

   重要　管理ポータルにこれら以外のフィールドが表示される場合は、使用しようとしている秘密鍵と証明書に向けて適切に指定していないことになります。表示されたすべてのフィールドに記入し、**[保存]** をクリックして成功すると、CA サーバの新しい秘密鍵と証明書の作成は完了です。

8. **[保存]** をクリックします。ローカル CA サーバの構成情報を保存する際、InterSystems IRIS は既存の証明書と秘密鍵を使用します(存在しない場合は SRL ファイルも作成します)。これで、次のような成功メッセージが表示されます。

```
Certificate Authority server successfully configured.
Using existing files: C:\pki\FileNameRoot.cer and .key
Certificate Authority Certificate SHA-1 fingerprint:
E3:FB:30:09:53:90:9A:31:30:D3:F0:07:8F:64:65:CD:11:0A:1A:A2
Confirmation email sent to: CAserver-admin@intersystems.com
```

新しい秘密鍵と証明書の作成の場合と同様に、この時点で、CA サーバは構成され、証明書署名要求（CSR）を処理できる状態になります。CA クライアントが CSR を作成すると、CA 管理者はこれについての電子メール通知を受け取ります。

CA サーバの再初期化

CA サーバとしてすでにインスタンスを構成している場合、CA の構成用のページには **[再初期化]** ボタンが表示されます。これを選択すると、以下のような影響があります。

- CA サーバのすべての構成情報が削除されます。
- 発行された証明書のすべての情報が破棄されます。
- CA で保留中のすべての証明書署名要求が破棄されます。

注釈　再初期化では、CA の秘密鍵や既存の証明書を含むファイルは削除されません。CA の既存の SRL ファイルも削除されません。実際、これらは依然として有効で、使用可能です。また、すでに署名済みの無効な証明書はレンダリングされません。

このボタンをクリックすると、CA を再初期することを確認するプロンプトが表示されます。再初期化すると、新しい CA サーバを構成することができます。

保留中の証明書署名要求の管理

認証局（CA）サーバが構成されると、CA サーバに関連する主なタスクは、潜在的な CA クライアントからの証明書署名要求（CSR）の管理となります。これには以下のようなタスクが含まれます。

- 証明書署名要求 (CSR) の処理
- 証明書署名要求 (CSR) の削除

要求を承認するように処理する場合、CA サーバは CA の秘密鍵を使用して署名された X.509 証明書を発行し、発行された証明書のシリアル番号を通知する電子メールを CA クライアントの技術担当者に送信します。ここでは、要求を削除 (すなわち拒否) することもできます。

このプロセスでの重要なステップは検証です。ここで CA 管理者は、偽装ができない通信を使用して、要求元の識別情報、技術担当者が要求された DN の証明書を保持する権限、および証明書が発行される目的を検証します(このために、CA サーバの管理者は、CSR と共に、潜在的な CA クライアントから受け取った連絡先情報を使用します)。

証明書署名要求 (CSR) の処理

要求の処理とは、CSR を証明書に変換することです。以下はその方法です。

1. 管理ポータルで、**[公開鍵インフラストラクチャ]** ページ (**[システム管理]** > **[セキュリティ]** > **[公開鍵インフラストラクチャ]**) に移動します。
2. **[公開鍵インフラストラクチャ]** ページの **[認証機関サーバ]** で、**[保留中の証明書署名要求の処理]** を選択します。これにより、CA が受け取った後、処理または削除が行われていない CSR のテーブルが表示されます。各 CSR の右には、**[プロセス]** および **[削除]** リンクが表示されます。
3. CA サーバの証明書と秘密鍵のファイルが含まれるメディアをマウントします(これは、CA サーバとして InterSystems IRIS インスタンスを構成する際にこれらのファイルを格納したメディアです)。
4. CSR を処理するには、**[プロセス]** をクリックします。これにより CSR の内容が表示されます。
5. 証明書を発行する前に、証明書の使用法を指定する必要があります。**[証明書の使用法]** ラジオ・ボタンの選択により、証明書が実行できる処理が指定されます。
   - **[TLS/SSL および XML セキュリティ]** – InterSystems IRIS 内のさまざまなセキュリティ機能を直接使用している CA クライアント用。
   - **[中間認証機関]** – InterSystems IRIS の他のインスタンスの CA として機能している CA クライアント用。
   - **[コード署名]** – コード署名を実行する CA クライアント用。
6. 重要　この手順では、偽装を防止する手段を使用して、潜在的な CA クライアントの技術担当者の識別情報を検証する必要があります。

   このページの手順で示したように、CSR を送信したインスタンスの指定された技術担当者に問い合わせる必要があります。組織のポリシーに従って、この担当者に電話で、または直接会って、確認します。
   - この担当者の識別情報
   - この担当者が、CA サーバ管理者が管理する CA によって署名された、上記所有者識別名を含む証明書を保持する権限
   - 上記の SHA-1 指紋が、証明書署名要求の送信時にレポートされた指紋と一致すること
7. 証明書の目的を指定し、技術担当者に関連する情報を検証したら、証明書を発行できます。これを実行するには、**[証明書の発行]** をクリックします。これにより、ページに **[認証機関の秘密鍵ファイルのパスワード]** フィールドが表示されます。
8. **[認証機関の秘密鍵ファイルのパスワード]** フィールドに、CA サーバの秘密鍵ファイルを解読するためのパスワードを入力します。InterSystems IRIS を使用して秘密鍵と証明書を作成した場合、これは **[認証機関の秘密鍵ファイルのパスワード]** に入力した値となります。他のツールを使用して秘密鍵と証明書を作成した場合、これはこの目的でそれらのツールに指定したパスワード (存在する場合) です。
9. **[完了]** をクリックすると、証明書が作成されます。成功すると、InterSystems IRIS には次のようなメッセージが表示されます。

```
Certificate number 31 issued for Certificate Signing Request
"Santiago Development Group"
```

10. CA サーバの証明書と秘密鍵を保持するメディアを取り外し、安全な場所に保存します。

これで、InterSystems IRIS により証明書が作成され、CA クライアントの技術担当者には電子メールで証明書がダウンロード可能であることが通知されました。CA クライアントの技術担当者は、証明書をクライアント・ホストにダウンロードできるようになりました。

証明書署名要求 (CSR) の削除

要求を削除する手順は以下のとおりです。

1. 管理ポータルで、**[公開鍵インフラストラクチャ]** ページ (**[システム管理]** > **[セキュリティ]** > **[公開鍵インフラストラクチャ]**) に移動します。
2. **[公開鍵インフラストラクチャ]** ページの **[認証機関サーバ]** で、**[保留中の証明書署名要求の処理]** を選択します。これにより、CA が受け取った後、処理または削除が行われていない CSR のテーブルが表示されます。各 CSR の右には、**[プロセス]** および **[削除]** リンクが表示されます。
3. CSR を削除するには、**[削除]** をクリックします。操作を確認するダイアログが表示されます。
4. 確認ダイアログで **[OK]** をクリックします。
5. 必要に応じてこれらのフィールドに入力し、**[保存]** をクリックします。

これで CSR が削除されます。

認証機関クライアント・タスク

認証機関 (CA) クライアントには、1 回限りの設定タスクがあります。これは以下のとおりです。

1. CA クライアントとしての InterSystems IRIS インスタンスの構成。ここでは、潜在的な CA クライアントへの CA サーバの場所を指定や、CA クライアントの技術担当者の連絡先情報の指定も行います。
2. CA 証明書のコピーの取得。これにより他の証明書の検証が可能となります。

設定タスクの後の CA クライアント・タスクは以下のとおりです。

1. CA サーバへの証明書署名要求 (CSR) の送信。ユーザの観点では、この際、識別名 (DN) およびその他の情報の指定が必要となります(インスタンスに複数の個別の証明書を持つ理由がある場合、これは繰り返し行われることもあります)。
2. さまざまな証明書のコピーの取得。これには、CA クライアントの独自の証明書に加え、CA サーバが発行した他の証明書も含まれます。

これらのタスクを実行すると、CA クライアントは、PKI を使用する必要のある処理を実行できるようになります。これらのタスクは、安全な接続の最後に行われるため、エンド・エンティティと呼ばれています。

注釈　これらは CA クライアントの管理者または技術担当者のタスクであるため、このセクションはこれらの担当者を対象とします。認証機関サーバ・タスクはこれとは異なり、CA サーバ管理者が担当します。

認証局クライアントとしての InterSystems IRIS インスタンスの構成

認証局 (CA) クライアントとして InterSystems IRIS インスタンスを構成する手順では、潜在的な CA クライアントに対する CA サーバの場所の指定や、CA クライアントの技術担当者の連絡先情報の指定も行います。その手順は以下のようになります。

1. 管理ポータルで、**[公開鍵インフラストラクチャ]** ページ (**[システム管理]** > **[セキュリティ]** > **[公開鍵インフラストラクチャ]**) に移動します。

2. [公開鍵インフラストラクチャ] ページの [認証機関クライアント] で、[ローカル認証機関クライアントの構成] を選択します。このページのフィールドは以下のとおりです。
   - [認証機関サーバ・ホスト名] – 必須。CA サーバのマシンの完全修飾名です(具体的には、これは InterSystems IRIS のインスタンスが実行されているマシンで、このインスタンスが CA サーバとして機能しています。CA クライアントとしてインスタンスを構成する前に、CA サーバとしてこれを構成する必要があります)。
   - [認証機関 Web サーバ・ポート番号] – 必須。CA サーバとして機能する InterSystems IRIS のインスタンスの Web サーバ・ポート番号です。
   - [認証機関サーバ・パス] – 必須。CA サーバの Web サービスのパスです。既定では、これは /isc/pki/PKI.CAServer.cls です(この値は、CA の Web サービスへのアクセスの際に、サーバのホスト名とポート番号と共に使用されます)。
   - [ローカル技術担当者] – CA クライアントの代わりに CA サーバに検証情報を提供する担当者です。この担当者について、以下の情報が必要です。
     - Name – 必須。CA クライアントの技術担当者の名前です。
     - [電話番号] – 担当者の電話番号です。CA 管理者が CA クライアントの証明書を発行する前に検証を行う際に、CA クライアントの技術担当者と連絡を取るための番号です。InterSystems IRIS では特定の検証方法を必要とせず、例えば直接会って確認することもできるため、電話番号は必須ではありません。
     - [電子メール・アドレス] – 担当者の電子メール・アドレスです。CA サーバがクライアントの CSR を処理し、証明書が発行されたことを通知する電子メールを、CA クライアントの技術担当者が受信するためのアドレスです。サーバ管理者が新たに発行された証明書についてクライアントの技術担当者に連絡する手段はほかにもあるため、電子メール・アドレスは必須ではありません。

3. 必要に応じてこれらのフィールドに入力し、[保存] をクリックします。

InterSystems IRIS は、“認証機関クライアントは正常に構成されました”などのメッセージにより成功したことを認識します。ここで、次のタスクは、CA サーバの証明書のダウンロードです。

認証局サーバへの証明書署名要求の送信

InterSystems IRIS のインスタンスが認証局 (CA) クライアントとして構成されると、CA サーバに証明書署名要求 (CSR) を送信できるようになります。表面的には、この際、識別名 (DN) およびその他の情報の指定が行われます。内部では、CA クライアントで以下のようないくつかのアクションが実行されます。

1. 公開鍵と秘密鍵のペアの生成。
2. 公開鍵と指定された DN を含む証明書署名要求 (CSR) の作成。
3. Web サービスを使用した CSR の CA サーバへの送信。

PKI インフラストラクチャでは、自動的に CSR を CA サーバに提供し、その送信を認識して、CA サーバの管理者に電子メール通知を送信します。この送信には、CA クライアントのローカル技術担当者の連絡先情報が含まれています。その後、CA 管理者は、偽装ができない通信を使用して、要求元の識別情報、技術担当者が要求された DN の証明書を保持する権限、および証明書が発行される目的を検証します。要求が承認されると、CA サーバによる証明書の作成などの処理が行われます。

CSR を CA サーバに送信する手順は以下のとおりです。

1. 管理ポータルで、[公開鍵インフラストラクチャ] ページ ([システム管理] > [セキュリティ] > [公開鍵インフラストラクチャ]) に移動します。
2. [公開鍵インフラストラクチャ] ページの [認証機関クライアント] で、[証明書署名要求の認証機関サーバへの送信] を選択します。このページのフィールドは以下のとおりです。

- **[ローカル証明書と秘密鍵ファイルのファイル名ルート(拡張子なし)]** – 必須。秘密鍵と証明書のファイル名の共通部分を指定します。したがって、ファイル名ルートとして `CAClient` を選択すると、秘密鍵は CAClient.key ファイルに保存され、証明書は CAClient.cer ファイルに保存されます。このフィールドで有効な文字は、英数字、ハイフン、およびアンダースコアです。ルートを文字列“cache”にすることはできません。
- **[認証機関の秘密鍵ファイルのパスワード]** および **[パスワード確認]** – オプション。CA クライアントの秘密鍵を暗号化および解読するためのパスワードです。
- **[所有者識別子]** – クライアント証明書のベアラを示す識別名 (DN) を定義する 1 つ以上の名前と値のペアのセットです。1 つ以上の属性の値を指定する必要があります。属性は以下のとおりです。
  - **[国]** – ISO 国コードを使用した、その国の 2 文字の国コードです。
  - **[州または県]** – 完全な州または県の名前です。
  - **[地域]** – 完全な市町村の名前です。
  - **[組織]** – 証明書が関連付けられている組織の名前です。規則により、この値は単に“InterSystems”や“InterSystems Corp”ではなく、“InterSystems Corporation”のように略さずに入力します。
  - **[組織単位]** – 部署など、その他の組織情報です。
  - **[共通名]** –“ドキュメント・テスト・クライアント”など、クライアントのわかりやすい名前です。

3. 必要に応じてこれらのフィールドに入力し、**[保存]** をクリックします。成功すると、InterSystems IRIS には次のようなメッセージが表示されます。

```
SHA-1 Fingerprint:
0C:DA:5F:06:04:C7:AE:64:61:9C:5C:29:35:49:88:0D:B6:E5:7D:98,
Certificate Signing Request "CAClient060412"
successfully submitted to the Certificate Authority at instance MyCA
on node CATESTCLIENT.CATESTDOMAIN.COM
```

   CSR の作成に成功した場合、InterSystems IRIS では以下のようなアクションが実行されたことになります。

   - 鍵のペアの作成。
   - 指定したファイル名ルートで管理者のディレクトリに秘密鍵を保存。
   - 公開鍵を含む CSR を作成し、指定したファイル名ルートで管理者のディレクトリ内のファイルに保存。
   - CA クライアントの構成プロセスの一部として指定されたホスト名、ポート、およびパスを使用して、その CSR を CA に送信。

   (プロセスが成功しなかった場合、InterSystems IRIS にはエラー・メッセージが表示されます。)

   ファイルが作成されたら、物理的に保護される暗号化されたリムーバブル・メディアに、この機密情報を保存することを強くお勧めします。

4. InterSystems IRIS に表示される SHA-1 指紋をコピーします。

   重要　　この情報は、後で、検証プロセスの一部として提供する必要があるため、なくさないでください。

5. この時点で、InterSystems IRIS を使用して CSR が作成され、送信されました。CA の管理者から問い合わせがあったら、最後の手順でコピーした SHA-1 指紋を提供してください。これにより管理者は証明書を作成します。この証明書は、″認証局サーバからの証明書の取得″ の説明に従って、取得することができます。

認証局サーバからの証明書の取得

認証局 (CA) クライアントは、構成されると、CA サーバに関連付けられた任意の証明書をダウンロードできます。これには以下が含まれます。

- CA サーバの証明書。

- それ自体の証明書。これは、CA クライアントが証明書署名要求 (CSR) を CA サーバに送信し、CA サーバがこの要求を承認すると使用可能となります。
- 任意の他の CA クライアントに対して CA サーバが作成した任意の証明書。

証明書を取得する手順は以下のとおりです。

1. 管理ポータルで、**[公開鍵インフラストラクチャ]** ページ (**[システム管理]** > **[セキュリティ]** > **[公開鍵インフラストラクチャ]**) に移動します。
2. **[公開鍵インフラストラクチャ]** ページの **[認証機関クライアント]** で、**[認証機関サーバからの証明書の取得]** を選択します。これにより、ダウンロードできる証明書のリストおよび、(ダウンロード済みかどうかに関係なく) 現在のインスタンスに対して発行されている証明書を表示するボタンが表示されます。通常は、CA サーバ証明書と自身の証明書の両方が必要です。このページから、いくつかのタスクを実行できます。
   - CA 証明書をダウンロードするには、**[認証機関の証明書の取得]** をクリックします。以下のような確認メッセージが表示されます。

     ```
     Certificate Authority Certificate (SHA-1 Fingerprint:
     E2:FB:30:09:53:90:9A:31:30:C3:F0:07:8F:64:65:CD:11:0A:1A:A2)
     saved in file "c:\intersystems\myinstnace\mgr\MyCA.cer"
     ```

   - CA が発行した任意の証明書 (CA クライアント自体の証明書を含む) をダウンロードするには、そのシリアル番号、CA クライアントのホストの名前 (**[ホスト名]** 列)、CA クライアントのインスタンスの名前 (**[インスタンス]** 列)、または証明書のルート・ファイル名 (**[ファイル名]** 列) で証明書を探すことができます。
   - 現在のインスタンスに対して発行された任意の証明書を表示するには、**[このインスタンスの証明書を表示]** をクリックします。これにより、証明書のテーブルが表示され、ここから証明書をダウンロードできます。ここには、**[シリアル番号]** 列と **[ファイル名]** 列のみがリストされます。
3. **[取得]** をクリックして証明書をダウンロードすると、InterSystems IRIS には以下のような確認メッセージが表示されます。

   ```
   Certificate number 74 (SHA-1 Fingerprint:
   45:E8:DE:0C:15:BF:A7:89:58:04:5E:68:2E:4D:BB:01:F5:90:94:97)
   saved in file "c:\intersystems\myinstance\mgr\IstanbulAcctsPayable.cer"
   ```

InterSystems IRIS が最初に管理者のディレクトリに証明書をダウンロードする際、一度それらの証明書がクライアント・ホストに置かれると、それらはどこにでも移動することができます。

デモ：データベース暗号化

InterSystems IRIS デモ：データベース暗号化

ここでは、InterSystems IRIS® データ・プラットフォームでデータベース暗号化を処理する方法について説明します。これは組織のセキュリティ戦略にとって重要な部分です。

ここでは、データベース暗号化の概要を示し、暗号化されたデータベースの作成に関連するいくつかの初期タスクの手順を説明します。このガイドの完了後には、キー・ファイルを作成し、そのキー・ファイルをアクティブ化し、それを使用してデータベースを暗号化します。これらのアクティビティは、既定の設定と機能のみを使用する設計になっているので、ユーザはトピックの範囲を外れた詳細を扱うことなく、機能の基本部分を十分に理解することができます（ただし実装の実行時には、こうした詳細が重要になる可能性があります）。データベース暗号化の完全なドキュメントは、"暗号化ガイド" を参照してください。

データベース暗号化はなぜ重要か

暗号化によって、機密情報や個人情報の不適切または不正な使用や開示を完全に防げるわけではありませんが、保存データの暗号化を確実に行うことで、情報のセキュリティ防御における重要な層が提供されます。データベース・レベルで暗号化を配備すると、情報保護の制御に新たなディメンジョンが追加されます。

さらに、機密情報や個人情報に関する多くの法規制では、データ処理組織に防御の最前線として暗号化の採用を勧告または要求しています。ここには以下のような法規制が含まれます。

- 医療保険の相互運用性と説明責任に関する法律（米国 HIPAA 法）－ セキュリティ保護された保健情報を読み取り不可、解読不可、復元不可にすることを求めています。
- 米国マサチューセッツ州法（201 Code of Massachusetts Regulations（CMR）17.00）－ 送信中と保存済みの個人情報を暗号化することを求めています。
- 米国ニューヨーク州の規制（New York 23 New York Codes, Rules and Regulations（NYCRR）Part 500）－ 非公開情報を処理する金融機関やその他の関係組織は、データ保護の 1 つの手段として暗号化を利用する必要があります。
- EU 一般データ保護規則（GDPR）－ 安全保護において暗号化を保護制御として考慮に入れることを求めています。
- イタリアの個人データ保護に関する法律（PDPC）－ 最低限のデータ・セキュリティ対策に関する技術的な規定のセクション 24 で、健康と性生活を開示するデータの処理を暗号化することを求めています。
- オーストラリアの個人情報保護法（APP）の Principle 4 － 強固な暗号化の実装により、個人情報の保護に必要なプライバシー強化テクノロジに取り組みます。
- 日本の経済産業省（METI）ガイドライン － 暗号化されていない個人情報または機密情報の侵害が発生した場合、規制調査を実施する必要があります。個人情報の保護に関する法律（APPI）の第 20 条では、個人情報取扱事業者が情報の漏えい、紛失、またはき損の防止に対する責任を負う規定になっているからです。

こうした規制要件の多くは、急激に一般化している現象であるデータ漏えいに重点を置いていますが、現在のフレームワークは、ロールベースのアクセス、パスワード保護、侵入検出、データ損失防止、ロギング/監査、および暗号化などの適切なセキュリティ制御によってリスクに対処する義務を組織に負わせています。暗号化それだけでは、すべての必須要件が対処されませんが、セキュリティ基盤を提供するものです。データベース・レベルで暗号化を行うと、攻撃者はシステムやファイル領域へのアクセス権だけでなくデータベースへのアクセス権も得る必要が生じるため、保護機能が強化されます。この追加の層によって、組織やその顧客、すべての関係者に保証がもたらされます。

InterSystems IRIS によるデータベース暗号化の使用法

データベース操作に関連するアクティビティでは、InterSystems IRIS の暗号化と解読のプロセスはユーザに対して透過的です。エンド・ユーザまたはアプリケーション開発者の視点からは、アプリケーションは単純に通常のアクティビティを実行し、データがディスク上で自動的に暗号化されます。システム管理者の視点からは、データ暗号化が行われたこと

を確認するいくつかの単純なタスクが存在します。これらのタスクの実行後、再びアクティビティは非表示で実行されます。

さらに、こうしたアクティビティは最小限のプロセッサ時間しか使用しないので、アプリケーションに目立った影響が出ることはほとんどありません。また、当社のデータベースの構築方法により、これらのアクティビティは高度に最適化されています。

暗号化と解読には、米国政府の Advanced Encryption Standard (AES) が Cipher Block Chaining (CBC) モードで使用されています (単純に AES CBC と呼ばれています)。InterSystems IRIS は、AES CBC の 128、192、256 ビットのすべての法定キー・サイズをサポートしています。

InterSystems IRIS は、使用可能な最速の実装を使用して暗号化と解読を実行します。暗号化と解読では、使用可能な場合は常にプロセッサベースの命令セットとそれ本来の効率性が活用されます。最新の Intel および IBM POWER8 プロセッサはこうした命令を装備しています。InterSystems IRIS はこうした命令を自動的に検出して使用するので、ユーザがアクションを起こす必要はありません。Intel プロセッサでは、これは Advanced Encryption Standard New Instructions (AES-NI) に該当します。IBM では AES VMX 命令セットになります。

データベース暗号化キーは、Key Management Interoperability Protocol (KMIP) をサポートするキー管理サーバか、データベース・キーの暗号化コピーを格納するファイル内に保存できます。どちらにも独自の利点があります。

- KMIP は、クライアントがキー管理システムと通信するための OASIS 標準プロトコルです。KMIP サーバは、特殊なハードウェア・アプライアンスか、キー管理ソフトウェアを実行する一般目的のサーバにすることができます。
- データベース暗号化キーのファイル保存時に、インターシステムズは複数層の AES Key Wrap アルゴリズムを使用してキーを暗号化し、個々の管理者のキー暗号化キーは PBKDF2 アルゴリズムを使用して生成されるので、ディクショナリおよび総当り攻撃は実行不可能です。

留意すべき重要な点は、データベース暗号化はセキュリティ戦略に不可欠な部分ではあるものの、それだけではセキュリティの脆弱性に対処できないことです。伝送中のデータ保護など、その他のツールも非常に重要です。データベース暗号化が、InterSystems IRIS でデータ保護のために提供する一連のツールの一部であるのはこのためです。以下はその概要です。

- 政府規格のサポート – データベース暗号化に対する FIPS 140-2 (Federal Information Processing Standards) への準拠が検証されたライブラリを使用するように InterSystems IRIS を構成できます。これは Red Hat Linux で使用できます。
- 選択したデータ要素の保護 – データ要素暗号化と呼ばれるこの機能は、レコードの選択した部分 (クレジット・カード番号や社会保障番号など) のみを暗号化できるプログラミング手法を提供します。
- 伝送中のデータ保護 – InterSystems IRIS は、最新バージョンの TLS (Transport-Layer Security) を使用して、伝送中のデータを保護します。TLS は、データ通信を保護するための業界標準のプロトコルです。
- サードパーティ認証のサポート – InterSystems IRIS はサードパーティのサイトのリソースを使用するための認証をサポートします。よく見られるのは、Web 上でサイトを使用するために Facebook や Google を介してログインする方法です。これは Open Authorization Framework バージョン 2.0 (OAuth 2.0) を使用しており、別の層を介した認証 (OpenID Connect) が含まれることもあります。

データベース暗号化を試してみる

InterSystems IRIS データベース暗号化は簡単に使用できます。以下の単純な手順によって、暗号化されたデータベースを設定するための基本手順を説明します。

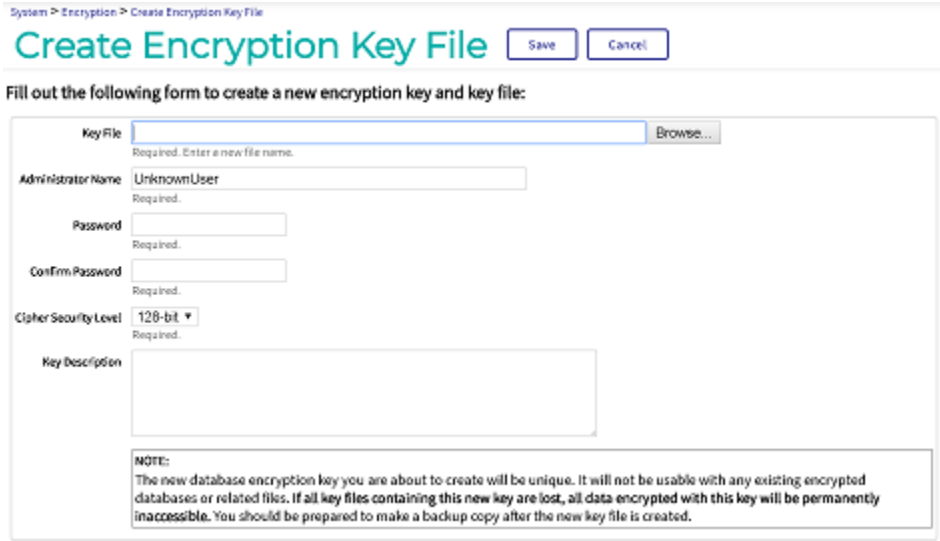
開始の前に

この手順を使用するには、InterSystems IRIS の稼働中インスタンスが必要です。選択肢としては、いくつかのタイプのライセンス付与されたインスタンスおよび無料の評価版インスタンスがあります。操作を実行するインスタンスをまだ用意できていない場合にインスタンスのタイプ別の導入方法の詳細を確認するには、″InterSystems IRIS の基礎 : IDE の接続″ の ″InterSystems IRIS の導入″ を参照してください。

暗号化キーの作成

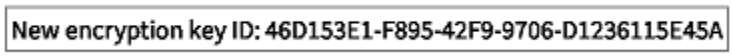
最初にキー・ファイルを作成します。ここでデータベース暗号化キーが自動生成されます。

1. "InterSystems IRIS の基礎：IDE の接続" で説明している該当のインスタンス用の URL を使用して、ブラウザ内でインスタンスの管理ポータルを開きます。
2. **[暗号化キーファイルを作成]** ページ (**[システム管理]** → **[暗号化]** → **[新しい暗号化キーファイルを作成]**) に移動します。



3. **[暗号化キーファイルを作成]** ページで、次の手順を実行します。
   a. **[キー・ファイル]** フィールドに、キー・ファイルの名前とパスを入力します。**[参照]** ボタンをクリックすると、既定でインスタンスの install-dir/**mgr** ディレクトリ (install-dir は InterSystems IRIS のインストール・ディレクトリ) (例えば、**C:¥InterSystems¥IRIS¥mgr¥testkeys.key**) で [ファイル選択] ダイアログが開きます。すべてのタイプのインスタンスでこのディレクトリを使用することも、ホストまたはコンテナのファイル・システム内の別の場所を選択することもできます。
   b. **[管理者名]**、**[パスワード]**、**[パスワード確認]** フィールドに、`testadmin` や `password` などの値を入力します。これは単なる実践例なので、開発環境で使用するパスワードを再利用しないでください。
   c. ページ上部にある **[保存]** ボタンを選択します。

C:¥InterSystems¥directory に **testkeys.key** というキー・ファイルが作成されました。このファイルにはデータベース暗号化に使用できるキーが含まれています。InterSystems IRIS には、次のように、ファイル内にあるキーを示すメッセージが表示されます。

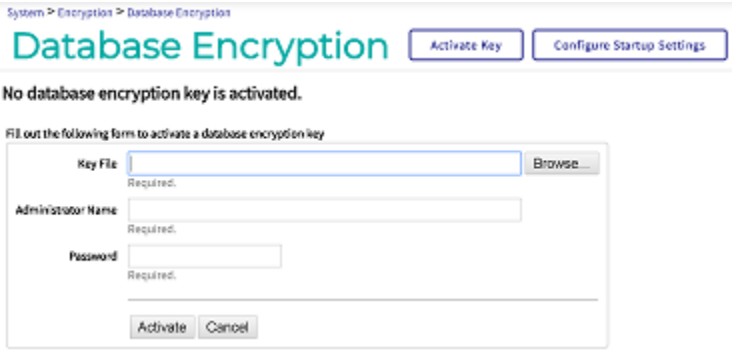


キー・ファイルおよびその初期キーの作成の詳細は、"キー・ファイルの作成" を参照してください。

暗号化キーの有効化

次に、作成したキーを有効化します。

1. 管理ポータルで、**[データベース暗号化]** ページ (**[システム管理]** → **[暗号化]** → **[データベース暗号化]**) に移動します。
2. [データベース暗号化] ページで、**[キーを有効にする]** ボタンを選択します。



3. [**キーファイル**] フィールドに、作成したキー・ファイルの場所を入力します (C:¥InterSystems¥IRIS¥mgr¥testkeys.key など)。
4. [**管理者名**] および [**パスワード**] フィールドに、指定した値を入力します (`testadmin` と `password`)。
5. [**有効化**] ボタンを選択します。

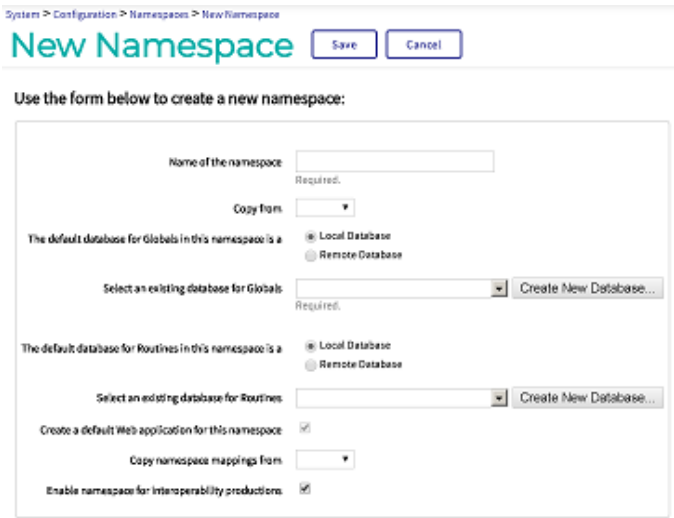
このページにキー ID が表示されます。

キーの有効化の詳細は、"キー・ファイルからのデータベース暗号化キーの有効化" を参照してください。

暗号化データベースの作成

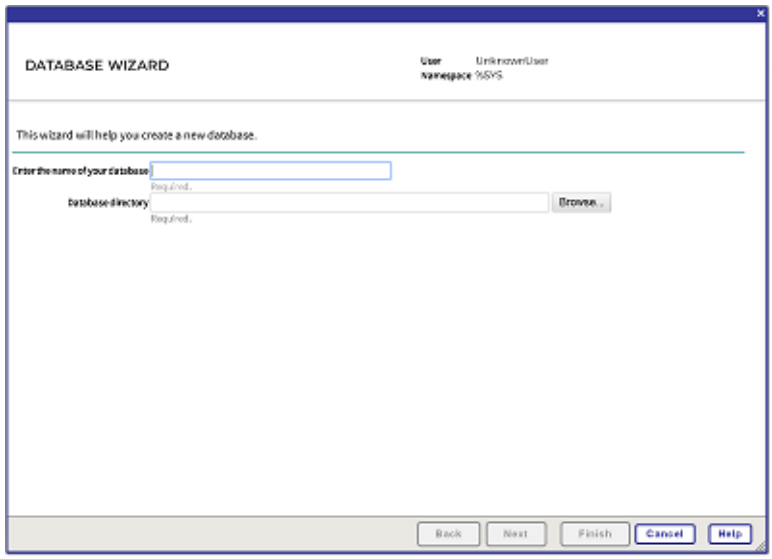
これで、暗号化データベースを作成できるようになりました。

1. 再び管理ポータルで、[**ネームスペース**] ページに移動します ([**システム管理**] → [**構成**] → [**システム構成**] → [**ネームスペース**])。
2. [**ネームスペース**] ページで、[**新規ネームスペース作成**] を選択します。[**新規ネームスペース**] ページが表示されます。



3. [**新規ネームスペース**] ページで、作成する暗号化データベースの名前を入力します (**ENCDB** など)。

4. [**グローバルに既存のデータベースを選択**] ドロップダウン・メニューの横の [**新規データベース作成**] ボタンを選択します。[**データベースウィザード**] が表示されます。



5. [**データベースウィザード**] の最初のページで、[**データベースの名前を入力してください**] フィールドに、作成するデータベースの名前 (**ENCDB** など) を入力します。データベースのディレクトリ (**C:¥InterSystems¥IRIS¥mgr¥ENCDB** など) を入力します。このページで [**次へ**] を選択します。
6. 次のページで、[**暗号化データベース**] の値を `No` から `Yes` に変更します。このページで [**完了**] を選択します。
7. [**新規ネームスペース**] ページに戻り、[**ルーチンに既存のデータベースを選択**] ドロップダウン・メニューで、作成したデータベースを選択します (**ENCDB** など)。
8. ページの上部にある [**保存**] ボタンを選択し、次に、結果のログの末尾の [**閉じる**] を選択します。

これで **ENCDB** という名前の暗号化データベースが作成されました。このデータベースはキー・ファイルの作成時に InterSystems IRIS によって作成されたキーを使用します。このデータベースは、暗号化されていないデータベースと同じように使用できます。InterSystems IRIS では暗号化と解読のすべての機能が非表示で行われるので、ユーザはすべての操作を通常の方法で実行することができます。データはすべて暗号化されます。

ネームスペースとそれに関連するデータベースの詳細は、″InterSystems IRIS システム管理ガイド″ の “InterSystems IRIS の構成” の章にある “ネームスペースの作成/変更” を参照してください。背景情報は、″サーバ側プログラミングの入門ガイド″ の “ネームスペースとデータベース” を参照してください。

暗号化されたデータの確認

暗号化データベースの作成後は、その他の暗号化されていないデータベースと同様にこれを使用できます。唯一異なる点は、データの保存方法です。暗号化データベースに保存されたデータと暗号化されていないデータベースに保存されたデータの違いを確認するには、以下の単純なテストを実行できます。

1. ″InterSystems IRIS の基礎：IDE の接続″ で説明している該当のインスタンスについての手順を使用して、InterSystems IRIS のターミナルを開きます。
2. 以下のコマンドを使用して、暗号化データベースのネームスペースに切り替えます。

```
%SYS>set $namespace="ENCDB"
ENCDB>
```

3. **ENCDB** ネームスペースで、次のコマンドを実行します。

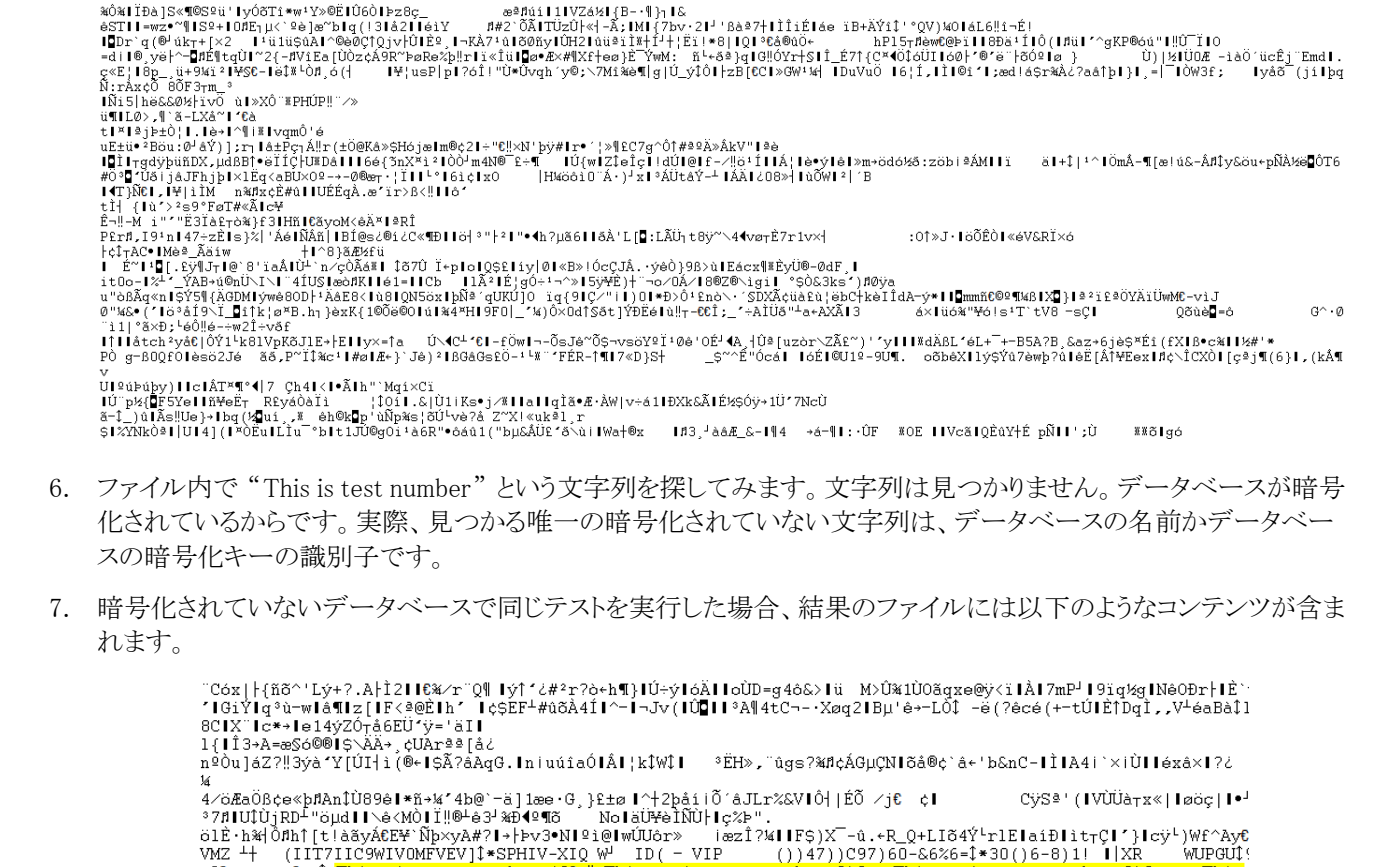
```
ENCDB>for i=1:1:1000 set ^x(i)="This is test number "_i
```

これにより、This is test number 22 というようなコンテンツを含む 1,000 個の永続変数が作成されます。

4. 成功したことを確認するには、1 つの変数の値を表示します。

```
ENCDB>w ^x(22)
This is test number 22
ENCDB>
```

5. データベース・ファイルを開くには、前のセクションでこれを作成したディレクトリ (C:¥InterSystems¥IRIS¥mgr¥ENCDB など) に移動して、データベース・ファイル IRIS.DAT を開きます。コンテンツは以下のように表示されます。



6. ファイル内で “This is test number” という文字列を探してみます。文字列は見つかりません。データベースが暗号化されているからです。実際、見つかる唯一の暗号化されていない文字列は、データベースの名前かデータベースの暗号化キーの識別子です。

7. 暗号化されていないデータベースで同じテストを実行した場合、結果のファイルには以下のようなコンテンツが含まれます。

スクリーン・ショットの最後の行に、ターミナルの変数セットの値が含まれています。

データベース暗号化に関係するその他の機能

InterSystems IRIS にはこの他にも、実装に重要な役割を果たすと思われる重要なデータベース暗号化機能が含まれています。

- KMIP – InterSystems IRIS では、インスタンスをホストするサーバとは別のサーバにキーを保存できます。こうしたサーバと通信するために、InterSystems IRIS は Key Management Interoperability Protocol (KMIP) をサポートしています。これにより InterSystems IRIS では、KMIP が提供するキー管理に対する標準手法の利点を得ることができます。
- キーの変更と暗号化の追加または削除 – データベースの暗号化キーを簡単に変更できます。必要とあれば、暗号化されていないデータベースの暗号化や、暗号化データベースの暗号化されていないコピーの作成も、簡単です。
- ディスク上の関連データの暗号化 – InterSystems では、最近のトランザクション・レコードを最新に保つ (つまり、ジャーナル・ファイルの) ために使用する一時的なキャッシュ・データベースやその他のディスク上のコンテンツを、簡単に暗号化できます。

- チップベースの暗号化 ー チップにより、アクティビティの一環として暗号化を実行できます。これにより動作が大幅に高速化します。InterSystems IRIS はこうしたチップの使用をサポートしています。チップベースの暗号化の詳細は、次のセクションを参照してください。

データベースの暗号化の詳細

インターシステムズでは、データベース暗号化を詳しく学習できるように、多数のリソースを提供しています。

- Encryption Awareness ー インターシステムズの暗号化テクノロジの概念を紹介するインターシステムズのオンライン学習インタラクティブ・コース。
- ″暗号化ガイド″ ー データベースの暗号化および関連機能に関するインターシステムズのドキュメント。
- ″データベース暗号化の FIPS 140-2 準拠″ ー InterSystems IRIS の FIPS 140-2 標準サポートに関するインターシステムズのドキュメント。

TLS

スーパーサーバでの TLS

TLS を使用するための InterSystems IRIS スーパーサーバの構成

InterSystems IRIS® データ・プラットフォームの複数のコンポーネント間の通信に TLS を使用するには、TLS を使用するように InterSystems IRIS スーパーサーバを構成します。そのための手順は以下のとおりです。

1. 管理ポータルのホーム・ページで、[SSL/TLS 構成] ページ ([**システム管理**]→[**セキュリティ**]→[**SSL/TLS 構成**]) に移動します。
2. [SSL/TLS 構成] ページで [**新規構成の作成**] を選択して、[**新規 SSL/TLS 構成**] ページを表示します。
3. [**新規 SSL/TLS 構成**] ページで、TLS サーバ構成を作成します。TLS 構成の作成の詳細は、"TLS 構成の作成または編集" を参照してください。
4. この構成を使用するようにスーパーサーバを設定します。詳細は、"スーパーサーバの管理" を参照してください。
5. TLS を適宜使用するようにクライアントを設定します ("TLS を使用するための InterSystems IRIS Telnet の構成" を参照してください)。

Telnet での TLS

TLS を使用するための InterSystems IRIS Telnet サーバの構成

Telnet クライアントからの TLS で保護された接続を受け入れるように InterSystems IRIS® を構成できます。そのためには、TLS を使用するように InterSystems IRIS Telnet サーバを構成します。

1. 管理ポータルのホーム・ページで、[SSL/TLS 構成] ページ ([**システム管理**]→[**セキュリティ**]→[**SSL/TLS 構成**]) に移動します。
2. [SSL/TLS 構成] ページで [**新規構成の作成**] を選択して、[**新規 SSL/TLS 構成**] ページを表示します。このページで、**%TELNET/SSL** の名前で TLS 構成を作成します。
3. Telnet サービス **%Service_Telnet** を有効にします。
   a. [**サービス**] ページ ([**システム管理**]→[**セキュリティ**]→[**サービス**]) の [%Service_Telnet] をクリックすると、その Telnet サービスの [**サービス編集**] ページが表示されます。
   b. [**サービス編集**] ページで、[**サービス有効**] チェック・ボックスがまだチェックされていない場合は、チェックを付けます。
   c. [**保存**] をクリックします。
4. 関連するスーパーサーバに対して TLS を有効にします。詳細は、"スーパーサーバでの TLS" を参照してください。
5. [**システムワイドセキュリティパラメータ**] ページ ([**システム管理**]→[**セキュリティ**]→[**システム・セキュリティ**]) で、[**Telnet サーバ SSL/TLS サポート**] 設定に [**有効**] を選択します。

TLS を使用するための Telnet クライアントの構成

InterSystems IRIS は、InterSystems IRIS Telnet クライアントとサードパーティの Telnet クライアントの両方からの TLS 接続を受け入れます。

TLS を使用するための InterSystems Telnet クライアントの構成

TLS 接続を使用するように InterSystems Telnet クライアントを構成できます。このプロセスは、複数の手順で構成されます。

1. Telnet サーバであるインスタンスで、TLS を要求するオプションの説明が含まれる前のセクションの手順に従って、そのインスタンスを構成します。
2. Telnet クライアントであるインスタンスで、"設定ファイルを使用した Windows クライアントからの接続" の手順に従って設定ファイルを構成します。

TLS を使用するためのサードパーティの Telnet クライアントの構成

InterSystems Telnet サーバに接続するようにサードパーティの Telnet クライアントを構成できます。必要な構成アクションまたは推奨される構成アクションは、使用中のソフトウェアおよび選択した暗号スイートによって異なります。次のガイドラインを目安にしてください。

- Telnet クライアントからサーバ認証が要求された場合、サーバは証明書を提供する必要があります。また、クライアントはサーバの証明書チェーンにアクセスできなければなりません。
- InterSystems IRIS Telnet サーバからクライアント認証が要求された場合、クライアントは証明書を提供する必要があります。また、サーバはクライアントの証明書チェーンにアクセスできなければなりません。
- InterSystems IRIS Telnet サーバからクライアント認証が要求された場合、クライアントは証明書と証明書チェーンを認証機関 (CA) に提供することもできます。クライアントが証明書を提供しなかった場合、認証は成功します。クライアントが不正な証明書、または証明書チェーンを提供した場合、認証は失敗します。

証明書や証明書チェーンが認証に使用される方法については、″必須証明書チェーンの確立″ を参照してください。

Python クライアントでの TLS

InterSystems IRIS との通信に TLS を使用するための Python クライアントの構成

Python クライアント・アプリケーションで InterSystems IRIS® データ・プラットフォームと通信するときに TLS が使用されるようにこのアプリケーションを構成できます。TLS を使用する Python 接続を確立するには、以下の手順を実行します。

1. ″TLS を使用するための InterSystems IRIS スーパーサーバの構成″ の説明に従って TLS を使用するようにスーパーサーバを構成します。
2. サーバ証明書を検証するための関連する CA 証明書がインストールされていることを確認します。
3. お使いのバージョンに基づいて Python クライアントを構成します。バージョン 4 では、SSL 構成に SSLDefs.ini ファイルを使用します。このファイルを構成する方法の詳細は、″設定ファイルを使用した Windows クライアントからの接続″ を参照してください。

V3

```
import ssl
import iris

context = ssl.SSLContext(ssl.PROTOCOL_TLS)
context.verify_mode = ssl.CERT_REQUIRED
cafile = "path/to/CACert.pem"
context.load_verify_locations(cafile)
context.load_cert_chain("path/to/Cert.pem", "path/to/Key.pem", "apasswordifany")

connection =
 iris.createConnection("127.0.0.1", 1972, "user", "_SYSTEM", "SYS", 10000, sslcontext=context)
...
connection.close()
```

V4

```
import iris
# On Windows lookup is based on address-port pair in SSLDefs.ini. -- GDConfig will be used
connection =
 iris.createConnection("127.0.0.1", 1972, "user", "_SYSTEM", "SYS", 10000, sslconfig=True)

# On Unix lookup is based on a provided configuration name instead of address-port pair. -- GDConfig2
 will be used
# connection =
 iris.createConnection("127.0.0.1", 1972, "user", "_SYSTEM", "SYS", 10000, sslconfig="GDConfig2")
...
connection.close()
```

以下に、上記の V4 コードで使用される Python クライアント構成の SSLDefs.ini ファイルの例を示します。

SSLDefs.ini

```
[IRIS]
Address=127.0.0.1
Port=1972
SSLConfig=GDConfig

[GDConfig]
TLSMinVersion=16
TLSMaxVersion=32
KeyType=2
VerifyPeer=0
CipherList=ALL:!aNULL:!eNULL:!EXP:!SSLv2
Ciphersuites=TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256
Password=apasswordifany
CertFile=path/to/Cert.pem
KeyFile=path/to/Key.pem
CAfile=path/to/CACert.pem

[GDConfig2]
TLSMinVersion=16
TLSMaxVersion=32
KeyType=2
VerifyPeer=0
```

```
CipherList=ALL:!aNULL:!eNULL:!EXP:!SSLv2
Ciphersuites=TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256
Password=apasswordifany
CertFile=path/to/AnotherCert.pem
KeyFile=path/to/AnotherKey.pem
CAfile=path/to/AnotherCACert.pem
```

注釈　このドキュメントの作成時点では、Python TLSv1.3 の暗号を変更することはできません。TLSv1.3 の場合は暗号スイートのみが使用され、値はこのリストと同じでなければなりません (順序は異なることがあります)。

TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256

TLSv1.3 より前のバージョンの場合は暗号リストのみが使用されます。

Java クライアントでの TLS

InterSystems IRIS との通信に TLS を使用するための Java クライアントの構成

Java クライアント・アプリケーションで InterSystems IRIS® データ・プラットフォームと通信するときに TLS が使用されるようにこのアプリケーションを構成できます。この通信はスーパーサーバ経由で行われるため、このスーパーサーバが TLS を使用するように設定する必要があります。詳細は、"TLS を使用するための InterSystems IRIS スーパーサーバの構成" で説明しています。Java クライアントは、JDBC またはオブジェクトのバインディングを使用して実装できます。

TLS を InterSystems IRIS で使用するように Java クライアントのアプリケーションを構成する手順は以下のとおりです。

1. クライアントでキーストアまたはトラストストアが必要であるかどうかを判断します。これは InterSystems IRIS サーバでクライアント認証を要求するかまたは必須とするか、サーバ認証が必須であるか、暗号スイートが使用されているかどうかなど、いくつかの要因によって変わります。詳細は、"キーストアおよびトラストストアが必要かどうかの判断" を参照してください。
2. これらの機能を提供するためのプロパティを持つ構成ファイルを作成します。詳細は、"クライアント構成の生成" を参照してください。
3. クライアント・アプリケーションのコードで、必要に応じて、クライアント構成の名前を指定します。名前を指定しなかった場合は、既定の構成情報が使用されます。詳細は、"クライアント構成の使用の指定" を参照してください。

キーストアおよびトラストストアが必要かどうかの判断

キーストアは、クライアントの秘密鍵、公開鍵証明書、および認証局 (CA) 情報のリポジトリの役割を果たします。この情報は、(1) InterSystems IRIS スーパーサーバによりクライアント認証が要求される場合、または (2) 使用している暗号スイートによりクライアント・キーのペアが要求される場合に必要です。

- InterSystems IRIS スーパーサーバがクライアント認証を必要としているかどうかは、関連する TLS 構成の **[SSL/TLS 構成を編集]** ページの **[相手証明書認証レベル]** フィールドに指定されている値によって判断されます。このフィールドの値が [ ] の場合、クライアントには証明書が必要です。[ ] の場合、サーバにより証明書があるかどうかがチェックされます。
- クライアントとサーバは使用する暗号スイートについて合意します。この暗号スイートにより、クライアント証明書、キーのペア、またはこの両方が存在するかどうかが判断されます。有効化されたサーバの暗号スイートは、関連するスーパーサーバの TLS 構成の **[SSL/TLS 構成を編集]** ページの **[有効な暗号スイート]** フィールドに指定されている値によって決定されます。クライアントで使用できる暗号スイートは、使用している Java のバージョンによって異なります。

クライアントが秘密鍵と証明書を持っている場合、これらはクライアントのキーストアに格納されています。このキーストアには、クライアントのルート CA 証明書とすべての中間 CA 証明書を保存できます。クライアントがサーバを認証するには、このサーバのルート CA 証明書と中間 CA 証明書が必要になります。これらは、クライアントのトラストストアに格納するか、またはクライアント証明書情報と共にキーストアに格納することができます。キーストアとトラストストアの詳細は、"Java Secure Socket Extension (JSSE) リファレンスガイド" の "キーストアとトラストストア" を参照してください。

クライアント構成の生成

Java クライアントの動作は、その構成で指定されているプロパティの値により異なります。この構成ではこれらの値を "構成ファイル" というファイルから取得します。具体的な値は、構成ファイルの既定値またはその構成固有の値のいずれかです。以下のセクションでは、構成ファイルの機能について説明します。

- 構成ファイル、構成、プロパティ、値、および既定
- Java クライアントの構成プロパティ
- 構成ファイルのサンプル
- 構成ファイルの名前付け

構成ファイル、構成、プロパティ、値、および既定

各構成ファイルでは、1 つ以上の構成で使用するプロパティの値を指定します。このファイルには、名前と値のペアの形式で、既定値と構成固有の値の両方が記述されています。一般的にこのペアでは、バージョン管理していないプロパティの名前に対しては既定値を指定し、バージョン管理しているプロパティの名前に対しては構成固有の値を指定します。

構成ファイルに記述されている構成定義が 1 つのみの場合、構成側ではバージョン管理していないプロパティを使用できます。ただし、関連付けられた name プロパティを持つことはできません。名前付き構成を使用しない場合は、名前を指定せずに構成を呼び出します (“クライアント構成の使用の指定”および“名前なしでの構成の指定”を参照)。

構成ファイルに複数の構成がある場合は、構成のバージョン番号 n を付加して name.n の形式とした name プロパティを指定することで各構成を定義します。構成のその他のプロパティ名では、name プロパティと同じバージョン番号が使用されます。したがって、名前の形式は propertyname.n となります。ここで、propertyname にはプロパティの名前、n には構成の番号が入ります。

構成ファイル内の定義では、大文字と小文字は区別されます。スペースは自由に使用できます。また、プロパティ定義の順番も自由です。

すべての構成で使用されるプロパティの既定値を指定するには、バージョン管理されていないプロパティ名とその値を次の形式で指定します。

```
propertyName = propertyValue
```

例えば、keyStoreType プロパティの既定値を pkcs12 に指定するには、以下の形式とします。

```
keyStoreType = pkcs12
```

このプロパティの既定値より優先する値を使用するには、次のようにバージョン管理しているプロパティ名を指定します。

```
keyStoreType.1 = jceks
```

1 つの構成ファイルに複数の構成定義がある場合は、そのバージョン番号順に構成を使用する必要があります。クライアント・アプリケーション・コードで参照する構成番号が連続していない場合はエラーになります。例えば、ある構成ファイルにバージョン管理された 3 つの **name** プロパティ、**name.1**、**name.2**、および **name.4** があるとします。この場合、**name.4** プロパティに関連付けられている構成は作成されず、このプロパティへの参照は失敗し、エラーが表示されます。

Java クライアントの構成プロパティ

以下のようなプロパティがあります。

- **cipherSuites** – サポートされている暗号スイートのコンマ区切りリスト。使用可能な暗号スイートは、使用しているマシン上の JRE (Java Runtime Environment) によって異なります。TLS ハンドシェイクの実行時に、サーバは、サーバとクライアントの両方でサポートされる最強の暗号スイートを選択します。(省略可)
- **debug** – デバッグ情報を Java **system.err** ファイルに記録するかどうかを表します。このプロパティには `true` または `false` を指定できます (既定値は `false`)。このプロパティの設定は、例外処理に影響を与えません。(省略可)
- **keyRecoveryPassword** – クライアントの秘密鍵へのアクセスに使用されるパスワード。これは秘密鍵のペアと同時に作成されたものです。(秘密鍵がパスワードで保護されていて、アプリケーション・コードが入力パラメータとして秘密鍵に渡されない場合に必須)
- **keyStore** – クライアントの秘密鍵と証明書情報を格納するためのファイル。また、キーストアには、一般にトラストストアに関連付けられるコンテンツも格納できます。(省略可)
- **keyStorePassword** – キーストアにアクセスするためのパスワード。(キーストアの作成時にパスワードを指定した場合には必須)
- **keyStoreType** – キーストア・ファイルの形式が指定されている場合はその形式。(省略可)

サポートされる形式は以下のとおりです。

- `jks` ― Java KeyStore。Java 独自の形式です。(既定)
- `jceks` ― Java Cryptography Extension KeyStore 形式。
- `pkcs12` ― Public Key Certificate Standard #12 形式。

- logFile ― Java がエラーの記録に使用するファイル。(省略可)
- name ― バージョン管理している Java クライアント構成の識別子(各 name プロパティはバージョン管理する必要があります。バージョン管理していない name プロパティは意味がなく、無視されます)。(省略可)

  構成ファイルで指定している構成が 1 つのみで、バージョン管理していないプロパティ名のみを使用している場合、name プロパティは必要ありません (“クライアント構成の使用の指定” を参照)。1 つの構成ファイルで複数の構成を指定する方法についての詳細は、″構成ファイル、構成、プロパティ、値、および既定″ を参照してください。
- protocol ― 接続に使用される TLS プロトコルのバージョン。(必須)

  サポートされている値には以下のものがあります。

  - `TLS` ― TLS プロトコルの任意のバージョン。TLS ハンドシェイクの実行時に、サポートされている最新バージョンのプロトコルがサーバによって選択されます。(既定)
  - `TLSv1` ― TLS バージョン 1。
  - `TLSv1.1` ― TLS バージョン 1.1。
  - `TLSv1.2` ― TLS バージョン 1.2。
  - `TLSv1.3` ― TLS バージョン 1.3。

- serverHostNameVerification ― 中間者攻撃を防止するために、この接続でサーバのホスト名の検証を実行するかどうか。このプロパティには true または false を指定できます (既定値は false)。(省略可)
- trustStore ― サーバのルート CA 証明書を格納するためのファイル。このファイルには、中間 CA の証明書も保持できます(この情報はキーストアに格納することもできます)。(省略可)
- trustStorePassword ―トラストストアにアクセスするためのパスワード。(キーストアの作成時にパスワードを指定した場合には必須)
- trustStoreType ― トラストストア・ファイルの形式が指定されている場合はその形式。(省略可)

  サポートされる形式は以下のとおりです。

  - `jks` ― Java KeyStore。Java 独自の形式です。(既定)
  - `jceks` ― Java Cryptography Extension KeyStore 形式。
  - `pkcs12` ― Public Key Certificate Standard #12 形式。

構成ファイルのサンプル

ここでは、Java クライアントで使用できる構成ファイルの例を示します。

```
debug = false
logFile = javatls.log
protocol = TLSv1.3
cipherSuites = TLS_AES_256_GCM_SHA384
keyStoreType = JKS
keyStore = keystore.jks
keyRecoveryPassword = <password>
keyStorePassword = <password>
trustStoreType = JKS
trustStore = truststore.jks
trustStorePassword = <password>
```

```
trustStoreRecoveryPassword = <password>

name.1 = IRISJavaClient1
keyStorePassword.1 = <password>
keyRecoveryPassword.1 = <password>
trustStorePassword.1 = <password>
trustStoreRecoveryPassword.1 = <password>

name.2 = IRISJavaClient2
protocol.2 = TLS
keyStoreType.2 = pkcs12
keyStore.2 = keystore.p12
keyStorePassword.2 = <password>
trustStore.2 = cjc1.ts
trustStorePassword.2 = <password>

name.3 = IRISJavaClient3
protocol.3 = TLSv1.2
debug.3 = true
cipherSuites.3 = TLS_RSA_WITH_AES_128_CBC_SHA
```

構成ファイルの名前付け

構成ファイルは、SSLConfig.properties という名前で保存するか、Java 環境変数 com.intersystems.SSLConfigFile の値をファイルの名前に設定します。コードは、現在の作業ディレクトリにあるファイルをチェックします。

クライアント構成の使用の指定

定義された構成は、サーバへの接続時にクライアント・アプリケーション・コードによって呼び出されます。これは、DriverManager オブジェクトまたは IRISDataSource オブジェクトの呼び出しで行うことができます。

DriverManager オブジェクトの使用法

DriverManager では、以下の手順を実行します。

1. Java Properties オブジェクトを作成します。
2. このオブジェクトの各種プロパティに値を設定します。
3. クライアントから InterSystems IRIS サーバへの接続のために、このオブジェクトを Java Connection オブジェクトに渡します。

接続で使用する情報を指定するには、まず構成ファイルから Properties オブジェクトを作成し、これに特定のプロパティの値を設定します。この処理を行うコードを最も簡単な形式で表すと、以下のようになります。

```
java.util.Properties prop = new java.util.Properties();
prop.put("connection security level", "10");
prop.put("SSL configuration name",configName);
prop.put("key recovery password",keyPassword);
```

以下はその説明です。

- 接続のセキュリティ・レベル 10 は、クライアントが TLS を使用して接続を保護しようとしていることを表します。
- configName は、Java クライアント構成の名前を値とする変数です。構成ファイルでは既定値のみを指定し、これらの既定値を 1 つの構成でのみ使用する場合は、この行を記述しないでください。詳細は、次の ″名前なしでの構成の指定″ を参照してください。
- keyPassword は、キーストアからクライアントの秘密鍵を抽出するために必要なパスワードです。

Properties オブジェクトが存在し、これに値が指定されると、最後の手順として InterSystems IRIS Java クライアントから InterSystems IRIS サーバへの接続にこのオブジェクトが渡されます。これは、DriverManager.getConnection メソッドへの呼び出しによって行われます。これを呼び出すための形式は次のとおりです。

```
Connection conn = DriverManager.getConnection(IRISServerAddress, prop);
```

ここで、IRISServerAddress は InterSystems IRIS サーバのアドレスを表す文字列、prop はこの文字列に渡される Properties オブジェクトです。

この呼び出しに成功すると、TLS で保護された接続が確立されます。通常、このセクションで説明したような呼び出しを含むアプリケーション・コードには、正常終了を確認するためのさまざまなチェック機構や、あらゆるエラーに対する保護が含まれます。InterSystems IRIS Java 接続の使用の詳細は、"InterSystems IRIS での Java JDBC の使用法" を参照してください。

IRISDataSource オブジェクトの使用法

IRISDataSource オブジェクトを使用する際は、オブジェクトを作成し、そのメソッドを呼び出して関連値を設定し、接続を確立します。メソッドは以下のとおりです。

- setConnectionSecurityLevel ー このメソッドは単一の引数 (接続のセキュリティ・レベル 10) を取ります。これは、クライアントが TLS を使用して接続を保護しようとしていることを表します。
- setSSLConfigurationName ー このメソッドは単一の引数 (Java クライアント構成の名前を値とする変数) を取ります。構成ファイルでは既定値のみを指定し、これらの既定値を 1 つの構成でのみ使用する場合は、この行を記述しないでください。詳細は、次の "名前なしでの構成の指定" を参照してください。
- setKeyRecoveryPassword ー このメソッドは単一の引数 (キーストアからクライアントの秘密鍵を抽出するために必要なパスワード) を取ります。

この処理を行うコードを最も簡単な形式で表すと、以下のようになります。

```
try{
   IRISDataSource ds = new IRISDataSource();

   ds.setURL("jdbc:IRIS://127.0.0.1:1972/TESTNAMESPACE");
   ds.setConnectionSecurityLevel(10);
   ds.setSSLConfigurationName(configName);
   ds.setKeyRecoveryPassword(keyPassword);

   Connection dbconnection = ds.getConnection();
}
```

プロパティの取得と設定に使用するメソッドの完全なリストは、"JDBC クイック・リファレンス" を参照してください。
com.intersystems.jdbc.IRISDataSource の JavaDoc は <install-dir>/dev/java/doc/index.html の下にあります。

名前なしでの構成の指定

構成ファイルに記述されている構成定義が 1 つのみの場合、構成側ではバージョン管理していないプロパティを使用できます。ただし、関連付けられた name プロパティを持つことはできません。

DriverManager オブジェクトを扱う場合、Properties オブジェクトでは構成ファイルにある既定値のみを使用します。このオブジェクトを作成するコードは、“SSL 構成名 ” キーの値を指定する呼び出しがないという点で通常のオブジェクト作成コードとは異なります。

```
java.util.Properties prop = new java.util.Properties();
prop.put("connection security level", "10");
prop.put("key recovery password",keyPassword);
```

IRISDataSource オブジェクトを扱う場合、名前のない構成を指定するには、単に setSSLConfigurationName の呼び出しを行わないようにします。

.NET クライアントでの TLS

InterSystems IRIS との通信に TLS を使用するための .NET クライアントの構成

InterSystems IRIS® データ・プラットフォームでは、.NET クライアントからの TLS 接続がサポートされています。

TLS を使用する .NET 接続を確立するには、以下の手順を実行します。

1. まだ TLS を使用するように InterSystems IRIS スーパーサーバを構成していない場合は、そのように構成して、.NET クライアントからの TLS 接続を受け入れることができるようにします。
2. .NET クライアントの TLS 構成を作成します。
3. サーバ証明書を検証するための関連する CA 証明書がインストールされていることを確認します。これらの場所は、現在のユーザの証明書ストア (Certificates - Current User¥Trusted Root Certification Authorities) です。
4. “インターシステムズ・データベースへの接続” の ”接続の作成” に記載されている接続文字列の形式に基づいて、サーバへの接続を確立します。サーバ、ポート、およびネームスペースの名前と値のペアのほか、SSL キーワードを追加してその値を `true` に指定します。例えば、TLS による保護を使用する接続を以下のような形式の接続文字列とします。

   ```
   IrisConnect.ConnectionString =
       "Server=localhost; Port=1972; Namespace=TESTNAMESPACE; SSL=true;"
       + "Password=SYS; User ID=_SYSTEM;";
   ```

   SSL キーワードに `true` を指定すると、TLS でクライアント・サーバ接続を保護できます。この場合は、.NET クライアントに対して InterSystems IRIS サーバが認証され、必要に応じてそのサーバに対してクライアントが認証されます。安全な接続が確立されると、InterSystems IRIS サーバではユーザ ID とパスワードのキーワードを使用して、.NET クライアントから接続するユーザの ID を認証します(相互認証に関しては、接続文字列では何も指定していません。接続文字列では、クライアント認証を要求または必要とするサーバを指定しているにすぎません)。

スタジオでの TLS

InterSystems IRIS との通信に TLS を使用するためのスタジオの構成

TLS 接続を使用するようにスタジオを構成できます。このプロセスは、複数の手順で構成されます。

1. スタジオからの接続を処理するスーパーサーバに対して、以下を行います。
   a. TLS 構成を設定します。このプロセスの詳細は、“TLS を使用するための InterSystems IRIS スーパーサーバの構成” を参照してください。
   b. TLS を有効にします。詳細は、″スーパーサーバの管理″ を参照してください。
2. スタジオが実行されている (TLS クライアントとして機能している) Windows マシンで、スタジオから TLS サーバ・インスタンスへの接続用の設定ファイルを構成します。このプロセスの詳細は、″設定ファイルを使用した Windows クライアントからの接続″ を参照してください。

.ini ファイルでの TLS と Windows

設定ファイルを使用した Windows クライアントからの接続

Windows を使用していて、スタジオ、ODBC、またはターミナルを TLS クライアントとして使用している場合、設定ファイルを使用して接続および構成を行うことができます。このメカニズムは、ホスト上に InterSystems IRIS® データ・プラットフォームのインスタンスがない場合でも利用できます。

設定ファイルを使用するには、以下の手順に従います。

1. サーバの認証機関 (CA) の証明書を取得します。それをディスクに保存して、保存場所を書き留めます (後で使用するため)。
2. "設定ファイルについて" の説明に従って、接続定義と構成定義を含むファイルを作成します。
3. ファイル SSLDefs.ini に名前を付けて、32 ビット共通プログラム・ファイル用のディレクトリの InterSystems¥IRIS ディレクトリに置きます。通常、このディレクトリは C:¥Program Files (x86)¥Common Files¥InterSystems¥IRIS¥ です。このディレクトリを探す必要がある場合は、Windows 環境変数 CommonProgramFiles(x86) (64 ビット Windows の場合) または CommonProgramFiles (32 ビット Windows の場合) の値を確認します。

ファイルを作成してこの場所に配置すると、ファイルにリストされているいずれかの接続と一致するホストおよびポートに接続したときに自動的にこのファイルが使用されるようになります。

注釈　設定ファイル (SSLDefs.ini) の使用には、以下の制限が適用されます。

1. この設定ファイルは、irisconnect.dll 実行可能ファイルまたは irisconnect64.dll 実行可能ファイル (それぞれ 32 ビット・マシン用と 64 ビット・マシン用) を使用する接続にのみ使用できます。他のメカニズム (ADO など) を使用する接続では、この設定ファイルは使用されません。
2. Windows クライアントから InterSystems IRIS への接続でこの設定ファイルが使用される場合、Kerberos 認証はサポートされません。

設定ファイルについて

設定ファイルには、TLS サーバへの接続の指定と、それらの接続で使用する TLS 構成の指定の両方が含まれます。TLS クライアントである Windows ホストごとに、すべての接続と構成が 1 つのファイルに記述されます。ファイルを作成するために必要な情報は、以下のとおりです。

- 設定ファイルの構文
- 接続プロパティ
- 構成プロパティ

設定ファイルの構文

設定ファイルには、1 つまたは複数の接続定義と 1 つまたは複数の構成定義が含まれます。

- 各定義は、接続または構成の識別子で始まります。識別子は、以下のように専用の行に括弧で囲んで記述します。

  ```
  [MyConfiguration]
  ```

  識別子には、以下のようにスペースや句読点を含めることができます。

  ```
  [MyOtherConfiguration, which connects outside of my local network]
  ```

- 各定義は、括弧で囲まれた次の識別子か、ファイルの終わりで終了します。
- 各定義には、複数のキーと値のペアが含まれます。そのすべてで以下の構文を使用します。

```
key=value
```

- キーと値のペアのグループで、接続定義または構成定義のプロパティを指定します。
- キーと値のペアの値は、引用符を付けずに記述します。

接続定義

各設定ファイルには 1 つまたは複数の接続定義が含まれます。その接続定義それぞれで TLS 接続のプロパティを指定し、その接続を TLS 構成と照合します。接続定義の最初の行は識別子で、括弧で囲んで記述します。識別子の後には、TLS サーバとそのサーバへの接続に関する情報を指定する複数の行が続きます。

Address

必須項目。TLS サーバのアドレス。IP アドレス、ローカル・ドメイン内の非修飾ホスト名、または完全修飾ホスト名を指定できます(注意 : Address と Port または TelnetPort の両方の値がクライアント・アプリケーションの接続先のサーバと一致する場合にのみ、クライアントは指定された構成を使用します)。

Port

必須項目。TLS サーバが接続を受け付けるポート番号(注意 : Address と Port または TelnetPort の両方の値がクライアント・アプリケーションの接続先のサーバと一致する場合にのみ、クライアントは指定された構成を使用します)。

TelnetPort

TLS で保護されている InterSystems Telnet 接続を受け入れる TLS サーバ上のポート番号。この値を指定しない場合、InterSystems IRIS Telnet を使用する接続で TLS はサポートされません(注意 : Address と Port または TelnetPort の両方の値がクライアント・アプリケーションの接続先のサーバと一致する場合にのみ、クライアントは指定された構成を使用します)。

SSLConfig

必須項目。この定義で指定されたサーバに接続する場合に、クライアントが使用する TLS 構成。各構成は、専用のセクションで定義します。

構成定義

各設定ファイルには 1 つまたは複数の構成定義が含まれます。その構成定義それぞれで TLS 構成のプロパティを指定します。TLS 構成の詳細は、“構成について” を参照してください。構成定義の最初の行は識別子で、括弧で囲んで記述します。構成識別子を接続定義の SSLConfig プロパティの値として記述すると、その構成を使用して接続動作が指定されます。識別子の後には、構成の各プロパティの値を指定する複数の行が続きます。

Protocols

非推奨。代わりに TLSMinVersion と TLSMaxVersion を使用します。

この構成でサポートする TLS プロトコルのバージョン。このプロトコルの各バージョンには、TLSMinVersion と TLSMaxVersion に挙げた数字が割り当てられています。複数のバージョンのプロトコルをサポートするよう指定するには、その値の合計を使用します。例えば、TLS v1.1 と TLS v1.2 のサポートを指定する場合は値 24 を使用します。

このプロパティは、管理ポータルの [TLS 構成] ページの **[プロトコル]** フィールドと同じです。

TLSMinVersion

v1.3 がサポートされる構成の場合は必須項目。この構成でサポートする TLS プロトコルの最小バージョン。サポート対象プロトコルの各バージョンを以下の数字で指定します。

- TLS v1 ー 4
- TLS v1.1 ー 8
- TLS v1.2 ー 16
- TLS v1.3 ー 32

このプロパティは、管理ポータルの [TLS 構成] ページの **[最小プロトコル・バージョン]** フィールドと同じです。

TLSMaxVersion

v1.3 がサポートされる構成の場合は必須項目。この構成でサポートする TLS プロトコルの最新バージョン。サポート対象プロトコルの各バージョンを以下の数字で指定します。

- TLS v1 ー 4
- TLS v1.1 ー 8
- TLS v1.2 ー 16
- TLS v1.3 ー 32

このプロパティは、管理ポータルの [TLS 構成] ページの **[最大プロトコル・バージョン]** フィールドと同じです。

VerifyPeer

必須項目。クライアントが接続先サーバの証明書を検証する必要があるかどうか。

- 0 ー 相手検証は不要です (実行されません)。すべての状況で接続が確立されます。
- 1 ー 相手検証が必要です。検証が成功した場合にのみ接続が確立されます。これが推奨値です。この値を選択する場合は、CAFile プロパティの値を指定する必要があります。

このプロパティは、管理ポータルの [TLS 構成] ページの **[サーバ証明書の検証]** フィールドと同じです。

VerifyHost

サーバの証明書の Common Name フィールドまたは subjectAlternativeName フィールドが接続定義で指定されているホスト名または IP アドレスと一致するかをクライアントが確認するかどうか。

- 0 ー 確認しません。
- 1 ー 確認します。

このプロパティと同等の機能は、管理ポータルにはありません。ただし、これは %Net.HttpRequest クラスの SSLCheckServerIdentity プロパティと同じタイプの確認です。

CipherList

TLS v1.2 以前を使用する接続がサポートされる構成の場合は必須項目。クライアントが暗号化およびハッシュのためにサポートする一連の暗号スイート。このプロパティの構文の詳細は、OpenSSL のドキュメントで ciphers コマンドを参照してください。

既定値は ALL:!aNULL:!eNULL:!EXP:!SSLv2 です。この値を使用することを強くお勧めします。InterSystems IRIS におけるこの構文の詳細は、"暗号スイート構文の有効化" を参照してください。

このプロパティは、管理ポータルの [TLS 構成] ページの **[有効な暗号スイート]** フィールドと同じです。

Ciphersuites

TLS v1.3 を使用する接続がサポートされる構成の場合は必須項目。クライアントが暗号化およびハッシュのためにサポートする一連の暗号。このプロパティの構文の詳細は、OpenSSL のドキュメントで ciphers コマンドを参照してください。

インターシステムズでは、この既定値 `TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256` を使用することを強くお勧めします。InterSystems IRIS におけるこの構文の詳細は、"暗号スイート構文の有効化" を参照してください。

CertFile

クライアントの信頼された認証機関 (CA) のファイルを含むファイルの絶対パスと名前。クライアントに CA がない場合は、このプロパティの値を指定しないでください。指定する場合、これは PEM 形式の X.509 証明書で、証明書チェーンを含めることができます。この値を使用する方法は、"必須証明書チェーンの確立" を参照してください(Windows の証明書のエクスポート ウィザードからエクスポートした証明書は、既定の DER でエンコードしたバイナリ X.509 ではなく、PEM でエンコードした X.509 形式とする必要があることに注意してください)。

このプロパティは、管理ポータルの [TLS 構成] ページの **[このクライアントの証明書を含むファイル]** フィールドと同じです。

KeyFile

構成の秘密鍵ファイルの絶対パスと名前。クライアントに秘密鍵がない場合は、このプロパティの値を指定しないでください。

このプロパティは、管理ポータルの [TLS 構成] ページの **[関連づけられた秘密鍵を含むファイル]** フィールドと同じです。

Password

構成の秘密鍵を解読するためのパスワード。パスワードを設定した秘密鍵を使用する場合、このプロパティは必須です。クライアントの証明書を使用しない場合、または秘密鍵にパスワードがない場合は、このプロパティの値を指定しないでください(秘密鍵がパスワードで保護されている場合、ここで値を指定しないと、InterSystems IRIS は秘密鍵を解読および使用できません)。

このプロパティは、管理ポータルの [TLS 構成] ページの **[秘密鍵パスワード]** フィールドと同じです。

KeyType

構成に秘密鍵と証明書がある場合に、構成の秘密鍵を保存する形式。

- DSA ー 1
- RSA ー 2

このプロパティは、管理ポータルの [TLS 構成] ページの **[秘密鍵タイプ]** フィールドと同じです。

CAfile

必須項目。サーバの信頼された認証機関 (CA) のファイルを含むファイルの絶対パスと名前。これは PEM 形式の X.509 証明書です。以下の点に注意してください。

- VerifyPeer 値を 1 に指定した場合は、この値を指定する必要があります。
- これは、接続するサーバの CA の証明書であり、自身が使用する CA の証明書ではありません。

このプロパティは、管理ポータルの [TLS 構成] ページの **[信頼済み認証局の証明書を含むファイル]** フィールドと同じです。ただし、ポータルとは異なり、`%OSCertificateStore` 文字列の使用はサポートしていません。

設定ファイルのサンプル

以下のサンプル・ファイルでは、2 つの接続と 2 つの構成を定義します。

```
[MyServer1 TLS to an InterSystems IRIS instance with TLS-protected InterSystems Telnet]
Address=myserver1
Port=57777
TelnetPort=23
SSLConfig=TLSConfig

[MyServer2 TLS to an InterSystems IRIS instance using TLSv1.2]
Address=myserver2.myexample.com
Port=57777
SSLConfig=TLSv1.2only

[TLSConfig]
TLSMinVersion=16
TLSMaxVersion=32
CipherList=ALL:!aNULL:!eNULL:!EXP:!SSLv2
Ciphersuites=TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256
KeyType=2
VerifyPeer=1
Password=
CertFile=c:\InterSystems\certificates\nopwclicert.pem
KeyFile=c:\InterSystems\certificates\nopwclikey.pem
CAfile=c:\InterSystems\certificates\cacert.pem

[TLSv1.2only]
TLSMinVersion=16
TLSMaxVersion=16
CipherList=ALL:!aNULL:!eNULL:!EXP:!SSLv2
KeyType=2
VerifyPeer=1
Password=
CertFile=c:\InterSystems\certificates\nopwclicert.pem
KeyFile=c:\InterSystems\certificates\nopwclikey.pem
CAfile=c:\InterSystems\certificates\cacert.pem
```

動作内容

重要　ここでは、インターシステムズ製品で設定ファイルを使用して TLS 接続を確立する方法について説明します。使用されているメカニズムについて説明することで、TLS 接続を作成する代替手段を示します。ここで説明する代替手段ではなく、上述の標準アプローチを使用することをお勧めします。

InterSystems IRIS では、以下のように設定ファイルを使用します。

1. TLS 接続を確立しようとすると、InterSystems IRIS TCP/IP クライアント接続ライブラリは、接続定義と接続構成を含む設定ファイルを探します。このファイルは、32 ビット・マシンでは **irisconnect.dll**、64 ビット・マシンでは **irisconnect64.dll** です。以下はその方法です。
   a. Windows レジストリで TLS 接続定義を確認します。
   b. レジストリに接続定義がない場合、ライブラリは設定ファイルに保存されている TLS 構成を見つけようとします。
   c. ISC_SSLconfigurations 環境変数が存在する場合、ライブラリは、この変数の値を設定ファイルのフル・パスおよびファイル名として使用します。

      注釈　ISC_SSLconfigurations 環境変数の値を定義する必要がある場合、管理者許可が必要になることがあります。

   d. ISC_SSLconfigurations 環境変数が存在しない場合、ライブラリは Windows 環境変数 CommonProgramFiles(x86) (64 ビット Windows の場合) または CommonProgramFiles (32 ビット Windows の場合) で指定された 32 ビット共通プログラム・ファイル・ディレクトリの下の **InterSystems¥IRIS** ディレクトリにある **SSLdefs.ini** ファイルを使用します。
2. 設定ファイルが見つかったら、ライブラリは、確立しようとしている接続に関連する接続定義を探します。

そのために、ファイルの各セクションで、確立しようとしている接続と一致する Address および Port プロパティを含むセクションを探します。セクションが見つかったら、そこにある SSLConfig プロパティの値を使用して、一致する TLS 構成セクションを探します。

3. 指定された TLS 構成セクションで、ライブラリは構成プロパティの値を使用して、サーバとの間で開始する接続のタイプを指定します。

データ要素暗号化

ミラーリングおよび TLS について

InterSystems IRIS® データ・プラットフォームによるミラーリングのサポートに関する一般情報は、"ミラーリング" を参照してください。

ミラー内でセキュリティを実現するために、TLS を使用するようにミラーのノードを構成できます。こうすると、1 つのノードから別のノードへの認証と、ノード間での暗号化された通信の両方が提供されます。フェイルオーバー・メンバの間で(および非同期メンバに対して)は、機密性の高いデータが受け渡しされるため、通信を暗号化して、ネットワークでデータが盗難も改ざんもされないようにすることをお勧めします。また、フェイルオーバー・メンバには、別の InterSystems IRIS システムに対してアクション (ジャーナル・ファイルの情報の要求や InterSystems IRIS の強制終了など) を実行するように ISCAgent に要求する機能があるため、ミラーのフェイルオーバー・メンバ (および対応する ISCAgent プロセス) の間でこのような通信を保護することは重要です。

注釈　フェイルオーバー・メンバがデータベース (またはジャーナル) の暗号化を使用している場合は、フェイルオーバー・メンバ間および任意の非同期メンバとの通信に TLS が必要です(特に、InterSystems IRIS は、どちらかのメンバが暗号化キーを有効化しているかどうかをチェックします。有効化している場合、インスタンスではユーザがミラーリングで TLS を有効化する必要があります)。データベース暗号化およびジャーナル・ファイル暗号化の詳細は、"暗号化ガイド" を参照してください。

(フェイルオーバー・メンバとして、または非同期メンバとして)ミラーリングに参加し、かつ TLS を使用するためには、インスタンスに 2 つの InterSystems IRIS TLS 構成 (サーバ・タイプとクライアント・タイプ) が必要です。またこれらの構成それぞれに、信頼された認証局によって発行された X.509 TLS 証明書が必要です。各ミラー・ホストが独自の証明書セットを持ち、この証明書には、証明書の共通名 (CN) コンポーネントに一意の識別子 (例えば、インスタンスの完全修飾ドメイン名 (FQDN) とメンバの InterSystems IRIS ノード名の組み合わせ) を含めることをお勧めします。これは、CN が証明書の識別名 (DN) のフィールドであり、一意の CN を作成すると、証明書の DN がメンバを一意に識別できるようになるためです。インスタンスのミラーリング構成を作成するには、次のセクションにある以下の手順を実行します。

TLS が有効になっている場合は、以下のアクションが行われます。

1. サーバ認証：クライアントがサーバに接続すると、サーバが自らを認証することが要求されます。この認証は、サーバの証明書の DN が、クライアントのミラー構成で構成されたシステムの DN と一致することを確認します。一致しない場合、クライアントは接続を切断します。
2. クライアント認証：サーバがクライアントからの接続を受け入れると、クライアントが自らを認証することが要求されます。この認証も、クライアントの DN が、サーバのミラー構成で構成されたシステムの DN と一致することを確認します。ここでも、一致しない場合、サーバは接続を切断します。
3. 暗号化：TLS プロトコルは、サーバの証明書を自動的に使用して、クライアントとサーバの間に暗号化されたチャンネルを確立し、このチャンネルを通過するデータがあればすべて暗号化され、保護されるようにします。

ミラーでは TLS を使用することを強くお勧めします。

TLS で非同期メンバを構成する場合の注意事項

ミラーで TLS を使用する場合は、そのミラーの TLS を有効化して各メンバの構成を作成 (後続のセクションを参照) するだけでなく、第 2 のフェイルオーバー・メンバまたは非同期メンバの構成時に特別な手順を実行する必要があります。詳細は、"第 2 のフェイルオーバー・メンバまたは非同期メンバを承認する (TLS ミラーのみ)" を参照してください。具体的には、フェイルオーバー・メンバごとに、**[ミラーモニタ]** ページで、**[メンバの X.509 証明書に DN としてリストされている ID]** フィールドに DN (識別名) を入力する必要があります。DN の値は、非同期メンバの **[非同期として参加]** ページ (**[システム管理]→[構成]→[ミラー設定]→[非同期として参加]**) の **[X.509 識別名]** フィールドからコピーできます (InterSystems IRIS は、非同期メンバの証明書の情報に基づいて **[X.509 識別名]** フィールドに生成し、このフィールドを手動で編集できないようにします)。

ミラーで TLS を無効にする場合の注意事項

既存のミラーで TLS を無効にする場合は、プライマリ・メンバで無効にします。

重要　　ミラーリングでは TLS を使用することを強くお勧めします。ミラーで TLS を無効にしないことを強くお勧めします。

ミラー用 TLS 構成の作成および編集

TLS をミラーで使用するには、(フェイルオーバーまたは非同期の) 各メンバは、`%MirrorClient` および `%MirrorServer` と呼ばれる TLS 構成のペアを使用します。ポータルでは、これらの構成の作成および編集が可能です。

注釈　　これらの構成は、ミラーで TLS が有効になっているときに、各メンバ上に既に存在している必要があります。

ミラー・メンバ用 TLS 構成の作成

この構成を作成する手順は以下のとおりです。

1. InterSystems IRIS のインスタンスのミラーリングがまだ有効になっていない場合は有効にします。そのためには、**%Service_Mirror** サービスの **[サービス編集]** ページを使用し、このページで **[サービス有効]** チェック・ボックスにチェックを付けます。このページには、以下の 2 つのパスのいずれかを使用して移動できます。
   - **[ミラー設定]** ページ (**[システム管理]→[構成]→[ミラー設定]**) で、**[ミラーサービスを有効にする]** を選択する。
   - **[サービス]** ページ (**[システム管理]→[セキュリティ]→[サービス]**) で、[%Service_Mirror] を選択する。

2. **[ミラー用 SSL/TLS 構成の作成]** ページに移動します。これには、以下のいずれかの方法を使用します。
   - [SSL/TLS 構成] ページ (**[システム管理]→[セキュリティ]→[SSL/TLS 構成]**) で **[ミラーのための構成を作成]** を選択します。
   - **[ミラーの作成]** ページ (**[システム管理]→[構成]→[ミラー設定]→[ミラーの作成]**) で、[SSL/TLS の構成] を選択します。

3. **[ミラーのための SSL/TLS 構成を作成]** ページでフォームの各フィールドに入力します。このページのフィールドは、**[新規 SSL/TLS 構成]** ページにあるフィールドの一部と同じです (*"TLS 構成の作成または編集"* を参照)。このページは、ミラーリングが自動的に有効になるサーバ構成とクライアント構成 (%MirrorClient と %MirrorServer) の両方を作成するため、**[構成名]**、**[説明]**、**[有効]** のいずれのフィールドもありません。また、秘密鍵パスワードに関しては、このページでパスワードの入力または置き換え (**[新規パスワード入力]**)、パスワードを使用しないことの指定 (**[パスワードクリア]**)、あるいは既存のパスワードをそのまま使用すること (**[そのままにする]**) の指定を行えます。

   両方の構成で同じ X.509 証明書が必要なため、このフォームの入力が完了すると両方の構成が同時に保存されます。このページの *"TLS 構成の作成または編集"* で説明しているフィールドは以下のとおりです。

   - [信頼済み認証局の証明書を含むファイル]

     注釈　　このファイルには、他のミラー・メンバに属する X.509 証明書の検証に使用できる証明書が含まれている必要があります。ファイルに複数の証明書が含まれている場合は、それらの証明書を正しい順序で (現在のインスタンスの証明書が最初になるように) 並べる必要があります。詳細は、*"必須証明書チェーンの確立"* を参照してください。

   - [このサーバの認証情報]：
     - **[関連づけられた秘密鍵を含むファイル]**
     - **[秘密鍵タイプ]**
     - **[秘密鍵パスワード]**
     - **[秘密鍵パスワード(再入力)]**

- [暗号方式設定]：
  - **[最小プロトコルバージョン]**
  - **[最大プロトコルバージョン]**

    [有効な暗号リスト (TLSv1.2以下)]
  - **[有効な暗号スイート (TLSV1.3)]**
  - **[DiffieHellmanビット数]**
- [OCSP 設定]
  - **[OCSP Stapling]**

フォームの入力が完了したら **[保存]** をクリックします。

ミラー・メンバの構成に関する一般情報は、"ミラーの作成" を参照してください。

ミラー・メンバ用 TLS 構成の編集

メンバの %MirrorClient 構成と %MirrorServer 構成の作成が済んでいる場合、それらの構成は **[ミラーのための SSL/TLS 構成を編集]** ページ (**[システム管理]**→**[セキュリティ]**→**[SSL/TLS 構成]** で **[ミラーのための構成を編集]** をクリック) で編集できます。このページには、前のセクションで説明した **[ミラー用 SSL/TLS 構成の作成]** ページと同じフィールドが表示されます。

ミラー・メンバの証明書に関する特別な考慮事項

TLS をミラーリングで使用する場合、%MirrorClient と %MirrorServer の構成では、同じ証明書と秘密鍵を使用する必要があります。したがって、両方の構成で使用されている証明書は、サーバ証明書としてもクライアント証明書としても使用可能である必要があります。

TLS のクライアントまたはサーバに固有の証明書エクステンションがあります。ミラーリングで使用されている証明書は、両方 (クライアントおよびサーバとして) の使用が可能である必要があるため、これらエクステンションのいずれかが証明書にある場合、クライアントとサーバの両方のエクステンションが必要になります。例えば、これは鍵用途および拡張鍵用途エクステンションで当てはまります。鍵用途エクステンションがある場合、以下の両方を指定する必要があります。

- デジタル・シグニチャの鍵用途 (クライアント用)
- 鍵暗号化の鍵用途 (サーバ用)

同様に、拡張鍵用途エクステンションがある場合、以下の両方を指定する必要があります。

- クライアント認証の鍵用途
- サーバ認証の鍵用途

両方のエクステンションがある場合、それぞれが両方の値を指定する必要があります。もちろん、エクステンションがどちらもない場合も有効です。

証明書で 1 つの値のみ (クライアントまたはサーバ) を指定した場合、ミラーリングの TLS 接続は、以下のようなエラーにより失敗します。

```
error:14094413:SSL routines:SSL3_READ_BYTES:sslv3 alert unsupported certificate
```

このエラーを解消する方法は、証明書の入手方法によって次のように異なります。

- 自己署名証明書を使用している場合、これらの条件に従った (OpenSSL ライブラリなどによる) 新しい証明書を作成します。

- 商用認証機関ツール (Microsoft Certificate Services など) を使用している場合、これらの条件に従った新しい証明書を作成し、このツールを使用して証明書署名要求 (CSR) に署名します。
- 証明書を商用認証機関 (VeriSign など) から購入した場合、証明書がこれらの条件に従うことの要求を CSR と共に含めます。

TCP デバイスでの TLS

TCP デバイスを使用して TLS を使用するための InterSystems IRIS の構成

ここでは、InterSystems IRIS® データ・プラットフォームの TCP 接続を利用して TLS を使用する方法について説明します。手順は以下のとおりです。

1. 必要な特性を指定する TLS 構成を作成します。
2. TCP 接続を開きます。または、TCP 接続を受け入れるソケットを開きます。
3. TLS を使用して接続を保護します。これは、接続やソケットを開くとき、またはその後で実行できます。

InterSystems IRIS TLS 機能の呼び出し方法は、InterSystems IRIS をクライアントまたはサーバとして使用しているか、最初に保護された TCP 接続を構築しているか、あるいは既存の接続に TLS を追加しているかによって異なります。

この章では、以下の項目について説明します。

- TCP 接続で TLS を使用するためのクライアントの構成
- TCP ソケットを使用して TLS を使用するためのサーバの構成

TCP 接続で TLS を使用するためのクライアントの構成

クライアントからの安全な接続を確立するには、以下のいずれかを実行します。

- クライアントから TLS で保護された TCP 接続を開く
- 既存の TCP 接続への TLS の追加

クライアントから TLS で保護された TCP 接続を開く

このシナリオでは、InterSystems IRIS はクライアントの一部であり、TCP 接続は最初から TLS を使用します。以下はその方法です。

1. 使用する構成が利用できることを確認します。InterSystems IRIS を最後に起動したときよりも前に構成が作成された場合、その構成は有効化されて使用できる状態です。それ以外の場合は、新しい構成を作成したり、既存の構成を編集したりできます。
2. TLS を使用して TCP 接続を開きます。

クライアントとして機能する InterSystems IRIS は、クライアント・アプリケーションを介してサーバに接続します。この接続では指定した構成を使用して、TLS 関連の動作を決定します。

TLS を使用して TCP 接続を開く

ここでは、TLS を使用する指定の接続を開き、特定のマシンやポート番号と通信します。以下はその方法です。

1. 接続しているデバイスを以下のように指定します。

   ObjectScript

   ```
   Set MyConn = "|TCP|1000"
   ```

   TCP の文字列は、これが TCP デバイスであることを指定しています。TCP 接続を開始する方法の詳細は、"TCP デバイスの OPEN コマンド" を参照してください。

2. 接続を開き、/TLS パラメータで、TLS の使用を指定します。

ObjectScript

```
OPEN MyConn:(SvrID:1000:/TLS="MyCfg")
```

以下はその説明です。

- MyConn は以前指定されたデバイスです。
- SvrID は、解決可能な DNS 名または IP アドレスの文字列です。
- MyCfg は、保存 (および有効化された) TLS 構成です。

この呼び出しでは、TLS を使用してポート 1000 のループバック・プロセッサ (つまり、ローカル・マシン) への TCP 接続を開きます。MyCfg 構成で指定した特性に従い、TLS が使用されます。

オプションとして、プライベート・キー・ファイルのパスワードを以下のように含めることができます。

```
OPEN MyConn:(SvrID:1000:/TLS="MyCfg|MyPrivateKeyFilePassword")
```

ここでは、すべての引数が示されており、MyPrivateKeyFilePassword が実際のパスワードになります。

重要　TLS を使用している TCP 接続を開くときにパスワードを含める機能は、リアルタイム・インタラクティブを使用する場合にのみ有効です。保護していない秘密鍵パスワードを永続的に保存することは絶対に避けてください。そのようなパスワードを保存する必要がある場合は、Security.SSLConfigs クラスの PrivateKeyPassword プロパティを使用します。

TCP デバイスを開く方法の詳細は、"TCP デバイスの OPEN コマンド・キーワードと USE コマンド・キーワード" を参照してください。

接続を一度確立すると、他の TCP 接続と同様に使用できます。

既存の TCP 接続への TLS の追加

このシナリオでは、TCP 接続が既に確立されていると仮定します。以下はその方法です。

1. 使用する構成が利用できることを確認します。InterSystems IRIS を最後に起動したときよりも前に構成が作成された場合、その構成は有効化されて使用できる状態です。それ以外の場合は、新しい構成を作成したり、既存の構成を編集したりできます。
2. TLS を使用して既存の TCP 接続を保護します。

TLS を使用した既存の TCP 接続の保護

ここでは、特定のマシンやポート番号への既存の接続に TLS を追加します。以下はその方法です。

1. 接続しているデバイスの名前を決定します。例えば、以下のコードを使用して接続が確立されているとします。

```
SET MyConn="|TCP|1000"
OPEN MyConn:("localhost":1000)
```

TCP の文字列は、これが TCP デバイスであることを指定しています。TCP 接続を開始する方法の詳細は、"TCP デバイスの OPEN コマンド" を参照してください。

2. /TLS パラメータで、TLS の使用を以下のように指定します。

```
USE MyConn:(::/TLS="MyCfg")
```

以下はその説明です。

- MyConn は以前指定されたデバイスです。
- MyCfg は、TLS 構成です。

オプションとして、プライベート・キー・ファイルのパスワードを以下のように含めることができます。

```
USE MyConn:(::/TLS="MyCfg|MyPrivateKeyFilePassword")
```

ここでは、すべての引数が示されており、MyPrivateKeyFilePassword が実際のパスワードになります。

重要　TLS を使用している既存の TCP 接続を開くときにパスワードを含める機能は、リアルタイムのインタラクティブを使用する場合にのみ有効です。保護していない秘密鍵パスワードを永続的に保存することは絶対に避けてください。そのようなパスワードを保存する必要がある場合は、Security.SSLConfigs クラスの PrivateKeyPassword プロパティを使用します。

TCP デバイスを開く方法の詳細は、"TCP デバイスの OPEN コマンド・キーワードと USE コマンド・キーワード" を参照してください。

接続に TLS セキュリティを追加しても、以前と同様にその接続を使用できます。

TCP ソケットを使用して TLS を使用するためのサーバの構成

クライアントからの安全な接続を必要とするソケットを有効にするには、以下のいずれかを実行します。

- この接続には TLS が必要であることを指定して TCP ソケットを開きます。
- 既存のソケットで TLS を使用する要件を設定します。

TLS で保護されたソケットの構築

このシナリオでは、InterSystems IRIS はサーバとして動作して、TCP ソケットは最初から TLS を使用します。以下はその方法です。

1. 使用する構成が利用できることを確認します。InterSystems IRIS を最後に起動したときよりも前に構成が作成された場合、その構成は有効化されて使用できる状態です。それ以外の場合は、新しい構成を作成したり、既存の構成を編集したりできます。
2. TLS の使用が必要な TCP ソケットを開きます。

このソケットでは、ソケットと接続するクライアントの TLS を使用する必要があります。クライアントがサーバへの接続を試行すると、サーバでは TLS を使用する接続をネゴシエートしようとします。これが成功すれば、接続を正常に使用でき、ネゴシエートされたアルゴリズムで通信が保護されます。失敗すると、クライアントで接続を使用できません。

TLS が必要な TCP ソケットを開く

TLS が必要なソケットを開くには、以下の手順を実行します。

1. 以下のように、接続を受け入れるデバイスを指定します。

   ObjectScript

   ```
   SET MySocket = "|TCP|1000"
   ```

   TCP の文字列は、これが TCP デバイスであることを指定しています。TCP 接続を開始する方法の詳細は、"TCP デバイスの OPEN コマンド" を参照してください。

2. 接続を開き、/TLS パラメータで、TLS の使用を指定します。

   ObjectScript

   ```
   OPEN MySocket:(:1000:/TLS="MyCfg")
   ```

オプションとして、プライベート・キー・ファイルのパスワードを以下のように含めることができます。

```
OPEN MySocket:(:1000:/TLS="MyCfg|MyPrivateKeyFilePassword")
```

この呼び出しでは、TLS を使用してポート 1000 の TCP ソケットを開きます。TCP デバイスを開く方法の詳細は、"TCP デバイスの OPEN コマンド・キーワードと USE コマンド・キーワード" を参照してください。

重要　TLS を使用している TCP 接続を開くときにパスワードを含める機能は、リアルタイム・インタラクティブを使用する場合にのみ有効です。保護していない秘密鍵パスワードを永続的に保存することは絶対に避けてください。そのようなパスワードを保存する必要がある場合は、Security.SSLConfigs クラスの PrivateKeyPassword プロパティを使用します。

既存のソケットへの TLS の追加

このシナリオでは、TCP ソケットへの接続が既に確立されていると仮定します。以下はその方法です。

1. 使用する構成が利用できることを確認します。InterSystems IRIS を最後に起動したときよりも前に構成が作成された場合、その構成は有効化されて使用できる状態です。それ以外の場合は、新しい構成を作成したり、既存の構成を編集したりできます。
2. TLS を使用して、ソケットへの既存の TCP 接続を保護します。

TLS を使用したソケットへの既存の TCP 接続の保護

ここでは、特定のマシンやポート番号のソケットへの既存の接続に TLS を追加します。以下はその方法です。

1. ソケットが開いているデバイスの名前を判断します。例えば、以下のコードを使用して接続が確立されているとします。

```
SET MySocket = "|TCP|1000"
OPEN MySocket:(:1000)
```

TCP の文字列は、これが TCP デバイスであることを指定しています。TCP 接続を開始する方法の詳細は、"TCP デバイスの OPEN コマンド" を参照してください。

2. /TLS パラメータで、TLS の使用を以下のように指定します。

```
USE MySocket:(::/TLS="MyCfg")
```

以下はその説明です。

- MySocket は以前指定されたデバイスです。
- MyCfg は、TLS 構成です。

オプションとして、プライベート・キー・ファイルのパスワードを以下のように含めることができます。

```
USE MySocket:(::/TLS="MyCfg|MyPrivateKeyFilePassword")
```

TCP デバイスを開く方法の詳細は、"TCP デバイスの OPEN コマンド・キーワードと USE コマンド・キーワード" を参照してください。

重要　TLS を使用している既存の TCP 接続を開くときにパスワードを含める機能は、リアルタイムのインタラクティブを使用する場合にのみ有効です。保護していない秘密鍵パスワードを永続的に保存することは絶対に避けてください。そのようなパスワードを保存する必要がある場合は、Security.SSLConfigs クラスの PrivateKeyPassword プロパティを使用します。

ソケットに TLS セキュリティを追加しても、以前と同様にソケットへの接続を使用できます。

Web ゲートウェイでの TLS

TLS を使用して InterSystems IRIS に接続するための Web ゲートウェイの構成

TLS を使用して、Web ゲートウェイと InterSystems IRIS® データ・プラットフォーム・サーバの間に暗号化されたセキュア・チャンネルを設定できます。これには、TLS 証明書、およびゲートウェイを表す秘密鍵が必要です。その後、ゲートウェイは InterSystems IRIS サーバ (専用の証明書および秘密鍵を所有) との暗号化された接続を確立できるので、すべての情報はこの接続を通じて伝送されます。

注釈　Web ゲートウェイと InterSystems IRIS サーバとの間に Kerberos で保護された接続を設定する方法の詳細は、"Web ゲートウェイと InterSystems IRIS 間での Kerberos で保護された接続の設定" を参照してください。

以下はその方法です。

1. InterSystems IRIS システムの既定のスーパーサーバに関連付けられた TLS 構成が存在しない場合、"TLS 構成の作成または編集" の説明に従い、構成を作成してください。
2. システムの既定のスーパーサーバ構成ページ ([**システム管理**]→[**セキュリティ**]→[**スーパーサーバ**]) で、[SSL/TLS support level] に [**有効**] または [**必須**] を選択します。この設定の詳細は、"スーパーサーバの管理" を参照してください。
3. Web ゲートウェイの [**サーバ接続**] ページ ([**システム管理**]→[**構成**]→[**ウェブゲートウェイ管理**]) に移動します。
4. そのページの [**構成**] で、[**サーバ接続**] を選択します。
5. 次に、[**サーバ編集**] を選択して、[**実行**] をクリックします。Web ゲートウェイの構成ページが表示されます。
6. このページで、TLS を使用するように Web ゲートウェイを構成します。具体的には、[**接続セキュリティレベル**] フィールドで [SSL/TLS] を選択します。[**SSL/TLS プロトコル**] フィールドと [**SSL CA 証明書ファイル**] フィールドの値を指定する必要があります。他のフィールドは、他の設定に応じて必須の場合もあればオプションの場合もあります。[**相手証明書認証が必要**] にチェックを付けた場合、[**SSL 証明書ファイル**] と [**SSL 証明書キーファイル**] は必須です。SSL/TLS 秘密鍵ファイルを使用する場合は、[**SSL キータイプ**] の値も指定する必要があります。また、証明書ファイルまたは秘密鍵ファイルでパスワードを必要とする場合は、[**SSL 秘密鍵パスワード**] フィールドで以下のいずれかを指定する必要があります。
   - 秘密鍵パスワード (先頭に `{`、または末尾に `}` を使用することはできません)
   - 中括弧で囲んだオペレーティング・システム・コマンド (例 : `{sh /tmp/script.sh}`)。詳細は、"プログラムによるパスワードの取得" を参照してください。

   このページの各フィールドの詳細は、"Web ゲートウェイの動作と構成" の "サーバ・アクセスの構成" を参照してください。

相互 TLS (mTLS)

概要

InterSystems IRIS では、インスタンスと InterSystems Web ゲートウェイ間の認証に相互 TLS (mTLS) をサポートしています。相互 TLS (双方向 TLS またはクライアント認証 TLS とも呼ばれます) は、ネットワーク通信の認証プロセスを強化するセキュリティ・プロトコルです。デジタル証明書を使用してクライアントとサーバ (InterSystems IRIS の場合、これはそれぞれ Web ゲートウェイと InterSystems IRIS インスタンスです) 両方の ID を検証することで、クライアントとサーバの間に安全な接続を確立します。相互 TLS は標準的な TLS ハンドシェイク・プロセスに従いますが、クライアント認証用の追加の手順があります。クライアントが接続を開始し、サーバが証明書で応答すると、サーバは相互認証用にクライアントの証明書を要求します。クライアントは証明書をサーバに送信し、両者は終了メッセージを交換して、ハンドシェイクの完了を確認します。サーバとクライアントの証明書はどちらも、信頼された認証局 (CA) によって署名されている必要があります。

前提条件

InterSystems IRIS で相互 TLS を有効にするには、いくつかの前提条件があります。開始する前に、以下が存在することを確認してください。

- InterSystems IRIS と通信できる Web ゲートウェイ。Web ゲートウェイの設定方法の詳細は、"概要：システムの Web ゲートウェイの設定" を参照してください。
- 信頼された CA 証明書。
- InterSystems IRIS スーパーサーバの証明書と秘密鍵。スーパーサーバの詳細は、"スーパーサーバの管理" を参照してください。

必要な証明書を収集して Web サーバをインストールしたら、スーパーサーバの SSL/TLS 定義を構成する必要があります。これを行う方法の詳細は、"TLS の構成" を参照してください。**[クライアント証明書の検証]** が [リクエスト] または [必要] に設定されていることを確認します。その後、この TLS 定義を使用するようにスーパーサーバを構成します。このスーパーサーバのポートを書き留めます。

ここまでで、Web ゲートウェイが設定され、クライアント証明書検証付きの TLS を使用するようにスーパーサーバが構成されているはずです。

設定

Web サーバをインストールし、クライアント証明書検証付きの TLS を使用するようにスーパーサーバを構成したら、以下の手順に従って、Web ゲートウェイの mTLS 認証を設定します。

1. **%Service_WebGateway** で mTLS 認証をオンにします。このサービスの詳細は、"サービス" を参照してください。
2. 任意の方法を使用して、Web ゲートウェイの新しい証明書を生成します。**CN** (一般名) フィールドは、既存の InterSystems IRIS ユーザの名前です。信頼された CA に、この証明書に署名してもらいます。
3. この証明書を使用するように Web ゲートウェイを構成します。
   a. Web ゲートウェイ管理ページに対して認証します。詳細は、"Web ゲートウェイ管理ページへのアクセス" を参照してください。
   b. **[構成]** で **[サーバ接続]** に移動し、サーバ構成プロファイルを編集または作成します。
   c. **[スーパーサーバの TCP ポート]** フィールドが、先ほど TLS を使用するように構成したスーパーサーバのポートと一致することを確認します。
   d. **[接続セキュリティ]** で、**[接続セキュリティレベル]** を [SSL/TLS] に設定します。
   e. **[ユーザ名]** と **[パスワード]** の既存の値をクリアします。これらの値は、相互 TLS よりも優先されます。
   f. **[SSL 証明書ファイル]** フィールドに、手順 2 で生成した Web ゲートウェイ証明書へのパスを入力します。

g. **[SSL 証明書キーファイル]** フィールドに、手順 2 で生成した Web ゲートウェイの秘密鍵へのパスを入力します。

h. **[SSL CA 証明書ファイル]** フィールドに、スーパーサーバ証明書に署名した CA 証明書へのパスを入力し、Web ゲートウェイがスーパーサーバの証明書を検証できるようにします。

i. 該当する場合は、**[SSL/TLS 秘密鍵パスワード]** フィールドに、Web ゲートウェイの秘密鍵パスワードを入力します。

j. 構成を保存します。

重要　Web ゲートウェイのサーバ・アクセス設定の、**[ユーザ名]** と **[パスワード]** の既存の値を必ずクリアしてください。これらの値は、相互 TLSよりも優先されます。

ここまでで、**%Service_WebGateway** で mTLS 認証が有効化され、CA 署名証明書付きの TLS を使用するように Web ゲートウェイが構成されているはずです (**CN** フィールドは、既存の InterSystems IRIS ユーザの名前です)。

テスト

設定が完了したら、Web ゲートウェイを使用して相互 TLS 認証が成功するかどうかをテストできます。認証をテストするには、以下の手順に従います。

1. Web ゲートウェイ接続をすべて閉じます。
   a. Web ゲートウェイ・ポータルから、**[管理]→[システムステータス]** に移動します。
   b. サーバ接続をすべて閉じ、[更新] ボタンをクリックします。
2. サーバ接続をテストします。
   a. Web ゲートウェイ・ポータルから、**[管理]→[サーバ接続のテスト]** に移動します。
   b. 設定中に相互 TLS に構成したサーバ・プロファイル名を選択します。
   c. **[接続]** をクリックします。

サーバ接続に失敗した場合、監査データベースで **%Service_WebGateway** からの LoginFailure イベントを確認できます。Web ゲートウェイ証明書の **CN** フィールドが、既存の InterSystems IRIS ユーザのユーザ名と一致することを確認します。監査データベースにそのようなイベントがない場合、システムが LoginFailure イベントを監査していることを確認します。詳細は、"監査" を参照してください。

LoginFailure イベントに対して監査を有効にしているが、このイベントが表示されない場合は、Web ゲートウェイ管理ページの [イベント・ログ] に移動して、さらなるトラブルシューティングのためにログ・メッセージを確認します。SELinux が Enforcing モードになっている場合は、"SELinux に関する考慮事項" を参照してください。

SELinux に関する考慮事項

SELinux が Enforcing モードに設定されている場合、証明書の SELinux のコンテキストを正しく構成しなければ、スーパーサーバに接続するための相互 TLS 認証は失敗します。CA、スーパーサーバ、Web ゲートウェイ証明書および関連する秘密鍵のコンテキストと OS 権限が正しく設定されていることを確認する必要があります。

スーパーサーバ証明書のコンテキスト

スーパーサーバ証明書と信頼された CA 証明書が irisusr グループから読み取り可能であり、適切なコンテキストで構成されていることを確認します。**<IRIS-install-dir>/mgr/** で生成された場合、ファイルは元の親ディレクトリのコンテキス

トを継承するため、これらには既に正しいコンテキストが含まれています。そうでない場合は、ファイルを <IRIS-install-dir>/mgr/ に移動して、Linux コマンド行インタフェースで以下のコマンドを実行します。

```
# restorecon -vF <IRIS-install-dir>/mgr/<superserver_certificate.cer>
# restorecon -vF <IRIS-install-dir>/mgr/<superserver_private.key>
# restorecon -vF <IRIS-install-dir>/mgr/<trusted_CA_certificate.cer>
```

restorecon コマンドは、ファイルのコンテキストを親ディレクトリのコンテキストにリストアします。スーパーサーバの TLS 構成に、これらのファイルの正しいパスが含まれていることを確認します。

Web ゲートウェイ証明書のコンテキスト

さらに、Web ゲートウェイの秘密鍵、証明書、およびスーパーサーバ CA 証明書が Web サーバ・ユーザまたはグループから読み取り可能であり、適切なコンテキストで構成されていることを確認する必要があります。適切なコンテキストを設定するには、まず Web ゲートウェイの秘密鍵、証明書、およびスーパーサーバ CA 証明書を /etc/ssl/certs/ ディレクトリに配置します。次に、Linux コマンド行インタフェースで以下のコマンドを実行します。

```
# restorecon -vF /etc/ssl/certs/<WebGateway_certificate.cer>
# restorecon -vF /etc/ssl/certs/<WebGateway_private.key>
# restorecon -vF /etc/ssl/certs/<superserver_CA_certificate.cer>
```

restorecon コマンドは、ファイルのコンテキストを親ディレクトリのコンテキストにリストアします。Web ゲートウェイのサーバ・アクセス・プロファイルに、これらのファイルの正しいパスが含まれていることを確認します。

証明書チェーン

必須証明書チェーンの確立

証明書と鍵を使用する暗号スイートを使用して接続を正常に確立するには、クライアントは、サーバ独自の証明書から、中間証明書（もしあれば）を含め、信頼された認証局（CA）の発行する自己署名証明書までのサーバの証明書チェーンを検証できなければなりません。サーバがクライアント・ユーザを認証している場合、このサーバは、クライアント・ユーザ独自の証明書から、中間証明書（もしあれば）を含め、信頼された CA の自己署名証明書までのクライアント・ユーザの証明書チェーンも検証できなければなりません。

認証は双方向で行うこともできるので、証明書チェーンに対する要件は、クライアントとサーバではなく、検証を行うエンティティ（認証を要求する側）と検証されるエンティティ（認証される側）を対象にしています。

認証を可能にするには、以下の条件を満たす必要があります。

- 検証を行うエンティティは、検証されるエンティティ独自の証明書から信頼された CA の自己署名ルート証明書までの証明書チェーンを構成するすべての証明書へのアクセス権が必要です。このチェーンに含まれる証明書は、検証されるエンティティの証明書ファイル（ハンドシェイク・プロトコルの一部として送信されたもの）と検証を行うエンティティの信頼された CA 証明書ファイルの組み合わせから取得されます。
- 検証を行うエンティティの CA 証明書ファイルには、信頼された CA の自己署名ルート証明書が必要です。
- 検証されるエンティティ独自の証明書は、証明書ファイルの先頭エントリでなければなりません。
- すべての中間 CA 証明書が必要です。
- 証明書チェーンに含まれる証明書は、検証されるエンティティの証明書ファイルと、検証を行うエンティティの信頼された CA 証明書ファイルに分けることができます。ただし、以下の例で説明するように、各部分は連続する部分証明書チェーンでなければなりません。

以下が存在すると仮定します。

- “ICA1”という認証機関により署名された証明書を持つ検証されるエンティティ（“VE”）。
- 認証機関“ICA2”により署名された“ICA1”の証明書、および“RootCA”により署名された“ICA2”の証明書。
- 自己署名ルート証明書を持つ信頼された CA（“RootCA”）。

これらの証明書を、検証されるエンティティと検証を行うエンティティに正しく分類すると、以下のようになります。

テーブル D-1: 証明書の有効な分類方法

検証されるエンティティの証明書ファイルに含まれる証明書	検証を行うエンティティの信頼された CA 証明書ファイルに含まれる証明書
VE	ICA1、ICA2、RootCA
VE、ICA1	ICA2、RootCA
VE、ICA1、ICA2	RootCA

ただし、検証されるエンティティの証明書ファイルに VE と ICA2 を、検証を行うエンティティの信頼された CA 証明書ファイルに ICA1 と RootCert を分類することは有効ではありません。