TLS ガイド

Version 2023.1
2024-01-02

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

TLS ガイド
InterSystems IRIS Data Platform　Version 2023.1　2024-01-02

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

テーブル一覧

1

TLS について

Transport Layer Security (TLS) では、エンティティ・ペア間における通信が強力に保護されます。これによって、認証、データ整合性保護、およびデータ暗号化が可能です。TLS は Secure Sockets Layer (SSL) の後継機能です。

SSL は Netscape で 1990 年代半ばに開発されました。TLS は SSL 3.0 の標準化として作成され、TLS version 1.0 は 1999 年にリリースされました。InterSystems IRIS で利用可能な TLS の最新バージョンは 1.3 で、多くの場合 TLS v1.3 と呼ばれます。InterSystems IRIS® データ・プラットフォームでサポートされているバージョンの TLS の中で、使用可能な最新バージョンを使用することをお勧めします。

注釈　　インターシステムズのドキュメントでは、用語としての SSL/TLS と SSL が TLS と同じ意味を持っています。

TLS 接続ではクライアント・サーバ・モデルが使用され、2 つのエンティティが TLS ハンドシェイクによって接続を確立します。2 つのエンティティでハンドシェイクが完了した場合、それは以下が行われたことを意味します。

- クライアントがサーバを認証した。
- サーバにクライアント認証が必要な場合にこれが行われます (クライアントとサーバの両方が互いを認証する方式は相互認証として知られています)。
- クライアントとサーバはセッション・キーについて合意した(セッション・キーは、対称鍵アルゴリズムで使用するためのキーで、これによってエンティティではそれ以降の通信でデータを保護できます)。
- それ以降の通信は暗号化できる。
- それ以降の通信の整合性は検証できる。

クライアントとサーバの暗号スウィートでは、これらの処理がハンドシェイクの一部としてどのように行われるか、またこれらの処理が保護された接続に対してどのようにサポートされるかが指定されます。特に、通信相手の暗号スウィートでは、サポートしている機能とアルゴリズムが指定されます。クライアントが使用可能な暗号化セットを提案し、提案された中からサーバが 1 つを選択します (クライアントとサーバとの間で共通の暗号化がないと、ハンドシェイクは失敗します)。

ハンドシェイクを行うには、通常、TLS は公開鍵暗号化を使用します (ただし、Diffie-Hellman プロトコルなどの他の方法も使用できます)。公開鍵暗号化では、それぞれの通信相手 (クライアントまたはサーバ) には公開鍵と秘密鍵があります。秘密鍵は機密の値で、公開鍵は幅広く公開される値です。一般的に、公開鍵は証明書にカプセル化されます。この証明書には、名前、組織、場所、発行者の妥当性などの所有者の識別情報も格納されます。InterSystems IRIS では、TLS 構成 (詳細は、"構成について" を参照) により、証明書ファイル、秘密鍵ファイル、暗号スウィートのオプション・セットなど、TLS 関連値の名前付きセットが指定されます。

成功すると、ハンドシェイクではセッション・キーが作成され、以降の通信を保護するために使用されます。

InterSystems IRIS とアプリケーションは TLS とのさまざまな相互作用を必要としますが、一般的にエンドユーザにはそのような直接の相互作用がありません。例えば、ブラウザでは TLS を使用して、指定された Web サイトと安全な接続を確立します。その際、サイト (この場合、サーバ) が自らをブラウザに認証する (ブラウザのユーザにはこれはわかりません) 必要があります。ブラウザに表示される鍵アイコンは、TLS により接続が保護されていることを示すためのものです。

1.1 InterSystems IRIS での TLS のサポート

InterSystems IRIS では、TLS がサポートされており、以下のようないくつかの接続タイプが保護されます。

- InterSystems IRIS スーパーサーバと対話するさまざまなクライアント・アプリケーション (ODBC、JDBC、スタジオなど) からの接続。
- Telnet サーバと対話する Telnet クライアントからの接続。
- InterSystems IRIS インスタンスがクライアントまたはサーバである (または、InterSystems IRIS インスタンスが両端にある) TCP 接続と共に使用するための接続。
- ECP (エンタープライズ・キャッシュ・プロトコル) を使用する接続。TLS を ECP と併用する方法の詳細は、"アプリケーション・サーバのデータ・サーバへの接続の TLS によるセキュリティ保護" を参照してください。

InterSystems IRIS がサーバとして機能する場合、接続を受け入れて TLS の使用を確立します。InterSystems IRIS がクライアントとして機能する場合は、TLS を使用する必要のあるサーバに接続できます。どのような場合でも、いわゆる TLS 構成が使用されます。この構成によって、TLS 接続の一部としての InterSystems IRIS インスタンスの各種特性が指定されます。

1.2 自身の InterSystems IRIS のインスタンスでサポートされている TLS のバージョン

InterSystems IRIS のインスタンスで利用可能な TLS のバージョンは、以下の複数の要因によって異なります。

1. オペレーティング・システム (OS) バージョンで利用可能な OpenSSL ライブラリのメジャー・バージョン。このライブラリにより、オペレーティング・システムでサポートされる、使用可能な TLS プロトコルのバージョンが決まります。
2. オペレーティング・システムのベンダがサポート対象バージョンのプロトコルに対して設定している他の制約 (Ubuntu 20.04 の制約など)。
3. このバージョンの InterSystems IRIS に対応する TLS の最小サポート対象バージョン。このリリースでは、TLS v1.0 です。

コンテナの場合、TLS のサポート対象バージョンは、コンテナ・ホストのオペレーティング・システムとバージョンによって異なります。

重要　オペレーティング・システムのバージョンによってプロトコルが変わるため、同じバージョンの InterSystems IRIS の 2 つのインスタンスであっても同じバージョンの TLS プロトコルがサポートされない場合があります。すべてのプラットフォームでサポートされているバージョンは TLSv1.2 だけです。

OS	バージョン	OpenSSL のバージョン	TLS のバージョン	注
AIX	7.2	1.0.2	1.0、1.1、1.2	以下の "AIX 7.2 の TLS に関する注" を参照。
AIX	7.3	3.0	1.2、1.3	
Red Hat Linux	7	1.0.2	1.0、1.1、1.2	以下の "Red Hat Linux 7 の TLS に関する注" を参照。
Red Hat Linux	8	1.1.1	1.0、1.1、1.2、1.3	以下の "Red Hat Linux 8 の TLS に関する注" を参照。
SUSE Linux	すべて	1.1.1	1.0、1.1、1.2、1.3	
Ubuntu Linux	18.04	1.1.1	1.0、1.1、1.2、1.3	
Ubuntu Linux	20.04	1.1.1	1.2、1.3	以下の "Ubuntu Linux 20.04 と 22.04 の TLS に関する注" を参照。
Ubuntu Linux	22.04	3.0	1.2、1.3	以下の "Ubuntu Linux 20.04 と 22.04 の TLS に関する注" を参照。
Windows	すべて	1.1.1	1.0、1.1、1.2、1.3	以下の "Windows の TLS に関する注" を参照。

注釈　Oracle Linux のバージョンについての情報は、Red Hat Linux の類似バージョンを参照してください。

1.2.1 AIX 7.2 の TLS に関する注

AIX 7.2 では OpenSSL 1.0.2 ライブラリを使用するので以下に注意してください。

- OpenSSL 1.0.2 ライブラリでサポートされるのは SSLv3 ～ TLSv1.2 です。InterSystems IRIS は SSLv3 をサポートしないため、TLSv1.0 ～ TLSv1.2 のみのサポートになります。
- OpenSSL 1.0.2 ライブラリには SHA-3 が含まれないため、インターシステムズは SHA-3 の独自実装を提供しています。この実装は RSASHA3Sign 関数および RSASHA3Verify 関数と互換性がありません。これらの関数の呼び出しは <UNIMPLEMENTED> エラーを返します。

1.2.2 Red Hat Linux 7 の TLS に関する注

Red Hat 7 では OpenSSL 1.0.2 ライブラリを使用するため、以下に注意してください。

- OpenSSL 1.0.2 ライブラリでサポートされるのは SSLv3 ～ TLSv1.2 です。InterSystems IRIS は SSLv3 をサポートしないため、TLSv1.0 ～ TLSv1.2 のみのサポートになります。
- OpenSSL 1.0.2 ライブラリには SHA-3 が含まれないため、インターシステムズは SHA-3 の独自実装を提供しています。
  - この実装は RSASHA3Sign 関数および RSASHA3Verify 関数と互換性がありません。これらの関数の呼び出しは <UNIMPLEMENTED> エラーを返します。
  - この実装は FIPS 検証済みではありません。SHA-3 関数の呼び出しはすべて <FIPS RESTRICT> エラーを返します。

1.2.3 Red Hat Linux 8 の TLS に関する注

FIPS モードの場合、Red Hat Linux 8 では TLSv1.2 と TLSv1.3 のみがサポートされます。

1.2.4 Ubuntu Linux 20.04 と 22.04 の TLS に関する注

Ubuntu 20.04 と 22.04 では TLSv1.2 と TLSv1.3 のみがサポートされます。これらのバージョンの Ubuntu では TLSv1.0 と TLSv1.1 の使用が禁じられているからです。

1.2.5 Windows の TLS に関する注

Windows では OpenSSL は使用されないため、インターシステムズは OpenSSL 1.1.1 ライブラリを InterSystems IRIS ディストリビューションの一部として出荷しています。したがって、TLSv1.0 ～ TLSv1.3 がサポートされます。

2

構成について

InterSystems IRIS® データ・プラットフォームでは複数の構成をサポートできます。それぞれの構成が、TLS 関連値の名前付きセットを指定します。既存の構成はすべて起動時に有効になります。管理ポータルで新しい構成を作成すると、その構成は保存時に有効になります。TLS 構成を管理するページは、[SSL/TLS 構成] ページ ([**システム管理**]→[**セキュリティ**]→[SSL/TLS 構成]) です。

2.1 TLS 構成の作成または編集

TLS 構成を作成または編集するページは、[SSL/TLS 構成] ページ ([**システム管理**]→[**セキュリティ**]→[SSL/TLS 構成]) です。新しい構成を作成するには、[**新規構成の作成**] をクリックして [**新規 SSL/TLS 構成**] ページを表示します。既存の構成を編集するには、その構成の名前の右側にある [**編集**] をクリックします([**ミラーのための構成を作成**] をクリックすることで、ミラー・メンバの新しい構成セットも作成できます。ミラーリングおよび TLS の詳細は、“ミラーリングで TLS を使用するための InterSystems IRIS の構成”を参照してください)。

TLS 構成を作成または編集する場合、以下のフィールドを使用できます。

- [**構成名**] – 構成を識別するための文字列。構成名には、すべての英数字、および “|” 文字以外の句読点を使用できます。InterSystems IRIS スーパーサーバの構成を作成する場合、その構成名は必ず `%SuperServer` にします。このトピックの詳細は、“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。
- [**説明**] – 任意のテキスト。
- [**有効**] – キーの有効化の際にこの構成を利用可能とするかどうかの指定。
- [**タイプ**] – この構成の使用目的。[**クライアント**] または [**サーバ**] を選択します。既定値は [**クライアント**] です。クライアントはプロトコルの使用を開始し、サーバは最初の要求に応答します (InterSystems IRIS スーパーサーバではサーバ構成が使用されます。TLS クライアントではクライアント構成が使用されます)。このフィールドに選択する値は、以下によって決まります。
  - 次のフィールドが、[**サーバ証明書認証**] フィールドと [**クライアント証明書認証**] フィールドのどちらであるか。クライアント用の構成の場合、次のフィールドは [**サーバ証明書認証**] になります。このフィールドでは、クライアントの接続先サーバの証明書に求められる可能性のある認証を指定します。サーバ用の構成の場合、次のフィールドは [**クライアント証明書認証**] になります。このフィールドでは、サーバへの接続を試行するクライアントの証明書に求められる可能性のある認証を指定します。
  - [**信頼された証明書機関の証明書を含むファイル**] フィールドの動作。
- [**サーバ証明書の検証**] または [**クライアント証明書の検証**] – 構成で接続相手の証明書の検証が必要かどうかを指定します。

クライアント用の構成では、[**サーバ証明書認証**] を指定する必要があり、以下の使用可能な値がサポートされています。

- [**なし**] ー どのような状況でも続行します。
- [**必要**] ー 証明書の認証が成功する場合にのみ続行します。

サーバ用の構成では、[**クライアント証明書認証**] を指定する必要があり、以下の使用可能な値がサポートされています。

- [**なし**] ー サーバ側でクライアント証明書を要求せず、また必要としないことを示します。
- [**要求**] ー 証明書を提供する (または提供しない) クライアントを許可します。クライアントが証明書を提供しない場合、認証は続行します。クライアントが証明書を提供して認証に失敗すると、認証が失敗します。
- [**必要**] ー クライアントが証明書を提供する必要があることを示します。認証は証明書の認証によって決まります。

- [**信頼された認証機関の証明書を含むファイル**] ー この構成が信頼する 1 つまたは複数の認証機関 (CA) の X.509 証明書 (PEM 形式) が含まれているファイルのパスと名前。構成では、信頼された CA の証明書を使用して、接続相手の証明書を検証します。一般に、プロダクション・システムは、公的に利用可能な証明書を持つ商用認証機関からの証明書を使用します。

  このフィールドについては、以下の点に注意してください。

  - ファイルのパスは、絶対パスとして指定することも、<install-dir>`/mgr/` ディレクトリが基準になる相対パスとして指定することもできます。
  - Windows と macOS では、ローカル・オペレーティング・システムが提供する、信頼された CA 証明書を構成で使用するように指定できます。そのためには、文字列 `%OSCertificateStore` をこのフィールドの値として指定します。

    Windows では、InterSystems IRIS は Microsoft ルート証明書プログラムと互換性があり、このプログラムが Windows Update を使用して追加の証明書をオンデマンドでフェッチします。証明書の更新を構成する方法の詳細は、Microsoft の Web サイトで "Configure Trusted Roots and Disallowed Certificates" を参照してください。
  - [**クライアント証明書認証**] の値が [**なし**] のサーバ構成の場合、このフィールドは使用できません (相手認証が存在しないため)。
  - 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ります。
  - ミラーリングでは、独自の証明書を検証するために十分な情報も構成に含まれている必要があります。

  これらの証明書を使用する方法は、"必須証明書チェーンの確立" を参照してください。このような証明書のファイル名と、証明書チェーンの確認方法については、OpenSSL のドキュメントで verify コマンドを参照してください。

- [**このクライアントの認証情報**] または [**このサーバの認証情報**] ー ローカル構成の X.509 証明書および秘密鍵がファイルとして必要な場合に、これらを格納したファイル名。
  - [**このクライアントの証明書を含むファイル**] または [**このサーバの証明書を含むファイル**] ー この構成独自の X.509 証明書の場所。この値は、絶対パスと相対パスのいずれかとして、PEM エンコードする必要があります。証明書チェーンも含めることができます。これを認証に使用する方法については、"必須証明書チェーンの確立" を参照してください (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)。
  - [**関連づけられた秘密鍵を含むファイル**] ー 構成の秘密鍵ファイルを格納する場所。絶対パスまたは相対パスで指定します。

- [秘密鍵タイプ] – 秘密鍵の生成に使用するアルゴリズム。有効なオプションは、[DSA] (Digital Signature Algorithm)、およびアルゴリズム開発者の名前から名付けられた [RSA] (Rivest、Shamir、Adleman) です。
- [秘密鍵パスワード] – 構成の秘密鍵を暗号化および解読するための任意のパスワード。

  注釈　秘密鍵がパスワードで保護されていて、ここに値を入力しない場合、InterSystems IRIS は、秘密鍵および証明書の公開鍵が相互に対応することを確認できません。この結果、対応しない鍵が鍵のペアとして保存される可能性が生じます。

- [秘密鍵パスワード (確認)] – 入力したパスワードが目的の文字列であることを確認するために、パスワードを再度入力します。

· [暗号方式設定]:

- [最小プロトコル・バージョン] – この構成でサポートされる TLS バージョンのうち、最も古いバージョン。これは、インスタンスでサポートできるすべてのバージョンが表示されるドロップダウン・メニューで、既定値は [TLS v1.2] です。以下の注を参照してください。
- [最大プロトコルバージョン] – この構成でサポートされる TLS バージョンのうち、最新のバージョン。これは、インスタンスでサポートできるすべてのバージョンが表示されるドロップダウン・メニューで、既定値は [TLS v1.3] です。以下の注を参照してください。
- [有効な暗号リスト (TLSv1.2以下)] – クライアントとサーバ間の通信の保護に使用する一連の暗号 (TLS v1.2 以前のバージョンを使用している場合)。このトピックの詳細は、"サポートされる暗号構文" を参照してください。
- [有効な暗号化スウィート (TLSv1.3)] – クライアントとサーバ間の通信の保護に使用する一連の暗号 (TLS v1.3 を使用している場合)。このトピックの詳細は、"サポートされる暗号構文" を参照してください。
- [DiffieHellmanビット数] – (サーバ専用) Diffie Hellman 暗号で使用する鍵のサイズ (ビット数)。鍵の最小サイズはオペレーティング・システムによって異なります。[自動] オプションでは、ローカルのオペレーティング・システムで必要最小限のサイズ以上の鍵サイズが指定されます。Open Web Application Security Project (OWASP) では、最小鍵サイズとして 2048 ビットを推奨しています。

  注釈　"自身の InterSystems IRIS のインスタンスでサポートされている TLS のバージョン" で説明されているように、利用可能な TLS のバージョンは、使用している基礎の OpenSSL ライブラリのバージョンによって異なります。InterSystems IRIS は、使用されている OpenSSL ライブラリを確認し、関連する利用可能な TLS のバージョンのみを提示しようと試みます。ただし、InterSystems IRIS が特定できない方法でシステムが構成されていることがあります。例えば、Ubuntu 20.04 に付属する OpenSSL の構成は既定値から変更されていて、TLS 1.0 と TLS 1.1 が禁止されています。このような場合に備えて、InterSystems IRIS のエラー・メッセージとログは、OpenSSL の構成と InterSystems IRIS 構成との間に競合があるかどうかを判断するのに役立つように設計されています。予期しない動作が発生した場合は、インターシステムズのサポート窓口 (WRC) またはオペレーティング・システムのベンダまでお問い合わせください。

· [OCSP settings]:

- [OCSPステープリング] – 構成で OCSP ステープリングがサポートされているかどうかを示します。

  クライアントで OCSP ステープリングが有効になっている場合、そのクライアントは OCSP ステープリングを要求します。サーバがステープリングされた OCSP 応答を提供しないか、応答が検証に失敗した場合、ハンドシェイクは失敗します。サーバで OCSP ステープリングが有効になっている場合、サーバは、クライアントから要求を受信すると、ステープリングされた OCSP 応答を提供します。サーバ側の構成で OCSP ステープリングが有効になっている場合、応答の有効期限が近いか、サーバが要求を受信してキャッシュされた応答が期限切れであれば、サーバはステープリングされた OCSP 応答を更新します。

注釈　構成がクライアントであるのかサーバであるのか、および目的の機能により、必須フィールドは異なります。TLS 構成によっては必要ではないフィールドもあります。

構成の作成プロセスまたは編集プロセスを完了するには、このページの上部に表示される以下のボタンを使用します。

- **[保存]** ー 構成を保存および有効にして、ダイアログを閉じます。既存の構成の変更や作成する構成を保存します。
- **[キャンセル]** ー 既存の構成の変更や作成する構成を保存せずにダイアログを閉じます。
- **[テスト]** ー 有効な構成情報であるかどうかを確認します。構成のロールがクライアントの場合、このボタンを選択するとサーバ (URL でなく、ホスト名) やポート番号のプロンプトも表示されます。InterSystems IRIS ではそのサーバとのテスト接続を確立しようとします(サーバ構成の作成時、このボタンは使用できません)。

  注釈　構成にエラーがなくても、一部の TLS サーバには **[テスト]** ボタンで正常に接続できないことがあります。これは、接続テストでは TLS ハンドシェイクの後に HTTP 要求が実行されるためです。サーバがハンドシェイクの前に StartTLS メッセージを予期している場合 (LDAP、SMTP、FTPS、または別のプロトコルでの使用時など)、実際にはサーバとの TLS 接続に成功していても、テストは失敗します。

2.1.1 証明書に必要な情報

クライアントがサーバを認証する場合、クライアントには、サーバ独自の証明書から、サーバの信頼された CA 証明書まで、これらの間にあるものすべてを含む完全な証明書チェーンが必要です。

サーバ TLS 構成を設定するときに、サーバの信頼された CA 証明書がルート証明書ではない場合、問題があります。認証を適切に機能させるために、クライアントでは、サーバの個人証明書から信頼された自己署名 CA 証明書へ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を必要とします。このチェーンはサーバの証明書ファイル (ハンドシェイク時に送信されたもの) とクライアントの信頼された CA 証明書ファイルの組み合わせから得られます。信頼された自己署名ルート CA 証明書はクライアントの CA 証明書ファイルに入っている必要があります。また、サーバの個人証明書はサーバの証明書ファイルの先頭エントリでなければなりません。その他の証明書は、これらの 2 つの場所に分けることができます。クライアントがサーバに対して認証を実行するときには、同じ制約が逆に適用されます。

証明書の形式に関しては、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ります。ファイルの拡張子に関係なく、すべての証明書は PEM でエンコードする必要があります。

2.1.2 有効化された暗号スウィートの構文

構成で許可されるのは、有効化された暗号スウィートを使用する接続のみです。有効化された暗号スウィートを指定するには、以下のいずれかを実行できます。

- 個々の暗号スウィートのリストを提供する
- OpenSSL の構文を使用して、有効化/無効化する暗号スウィートを指定する

暗号スウィート名のリスト、および有効な暗号スウィートを指定するための構文は、どちらも openssl.org の ciphers(1) のマニュアル・ページで説明されています。この構文を使用すれば、構成に対してさまざまな機能やアルゴリズムを使用する場合の必要事項や禁止事項のガイドラインを指定できます。

InterSystems IRIS 構成で暗号スウィートの既定値は、`ALL:!aNULL:!eNULL:!EXP:!SSLv2` です。これは、コロンで区切られた文で以下のグループに分けられます。

- `ALL` ー eNULL 暗号以外のすべての暗号スウィートを含めます。
- `!aNULL` ー 認証を提供しない暗号を除外します。
- `!eNULL` ー 暗号化を提供しない暗号を除外します。
- `!EXP` ー 輸出承認済みアルゴリズム (40 ビットおよび 56 ビット) を除外します。
- `!SSLv2` ー SSL v2.0 暗号スウィートを除外します。

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。

2.1.3 TLS を使用する InterSystems IRIS クライアント・アプリケーションに関するメモ

一部の動作については、InterSystems IRIS スーパーサーバと対話するクライアント・アプリケーションのサポートに InterSystems IRIS インスタンスを使用できます。

TLS を使用して InterSystems IRIS スーパーサーバと対話するクライアント・アプリケーションを使用する場合は、構成について、次の点に特に注意してください。

- **[構成名]** – クライアントの名前には制限はありませんが、接続を構成するためにはこの情報は必須です。
- **[タイプ]** – インスタンスは TLS クライアントと共にサービスを提供するので、[タイプ] には **[クライアント]** を指定する必要があります。
- **[暗号スウィート]** – 指定された暗号スウィートは、サーバにより要求または指定されたものと一致する必要があります。

また、"必須証明書チェーンの確立" で説明しているように、クライアントとサーバは互いの証明書チェーンを検証できるように構成することも必要です。

2.2 構成の削除

TLS 構成を削除するページは、**[SSL/TLS 構成]** ページ (**[システム管理]→[セキュリティ]→[SSL/TLS 構成]**) です。構成を削除するには、構成名の右側にある **[削除]** をクリックします。ポータルからアクションの確認が求められます。

2.3 予約済みの構成名と必須の構成名

InterSystems IRIS では、特定の機能で使用するために、いくつかの TLS 構成名が予約されています。このような機能を使用するときには、この予約構成名を使用する必要があります。予約構成名は以下のとおりです。

- `%MirrorClient` – TLS クライアントとして動作する場合のミラー・メンバ用。ミラーリングおよび TLS の詳細は、"ミラーリングで TLS を使用するための InterSystems IRIS の構成" を参照してください。
- `%MirrorServer` – TLS サーバとして動作する場合のミラー・メンバ用。ミラーリングおよび TLS の詳細は、"ミラーリングで TLS を使用するための InterSystems IRIS の構成" を参照してください。
- `%SuperServer` – 他の InterSystems IRIS コンポーネントからの接続を受け入れる場合の InterSystems IRIS スーパーサーバ用。TLS を使用するようにスーパーサーバを構成する方法の詳細は、"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。
- `%TELNET/SSL` – TLS で保護された接続を受け入れる場合の Windows Telnet サーバ用。ミラーリングおよび Telnet の詳細は、"TLS に対する InterSystems IRIS Telnet サーバの構成" を参照してください。

重要　TLS が正しく機能するようにするには、ここに表示されているとおりに、大文字と小文字を正確に区別して各構成名を使用する必要があります。

2.4 プログラムによる TLS 構成の作成、編集、削除

TLS 構成をプログラムで管理するには以下を使用します。

- Security.SSLConfigs クラス
- 適用可能な構成マージ・アクション
  - CreateSSLConfigs
  - ModifySSLConfigs
  - DeleteSSLConfigs

3

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 `%SuperServer` という構成名の TLS サーバ構成を作成します (構成名の大文字と小文字は、ここに指定したとおりのものを使用します)。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. [**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 [**スーパーサーバSSL/TLSサポート**] フィールドで、[**有効**] を選択します。これは、スーパーサーバが TLS で保護された接続をサポートする (ただし、必須ではない) ことを表します。

   注釈　TLS で保護された接続を要求するようにスーパーサーバを構成する場合、まず TLS を有効にするように指定します。

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)。

4

TLS を使用するための InterSystems IRIS Telnet の構成

InterSystems IRIS® データ・プラットフォームには、TLS で保護された Telnet 接続を使用するためのオプションがいくつか用意されています。

4.1 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. [**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、[**スーパーサーバSSL/TLSサポート**] 設定と [**Telnetサーバ SSL/TLS サポート**] 設定の両方に [**有効**] を選択します。

4.2 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。

4.2.1 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。

4.2.2 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ウィ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。

5

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、暗号スウィ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。

5.1 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 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ウィ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 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、その InterSystems IRIS インスタンスの "%SuperServer" TLS 構成に関する **[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ウィートについて合意します。この暗号スウィートにより、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ウィートは、その InterSystems IRIS インスタンスの "%SuperServer" TLS 構成に関する **[SSL/TLS 構成を編集]** ページの **[有効な暗号化スウィート]** 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ウィートは、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。

5.2 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

5.2.1 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。

5.2.2 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ウィートのコンマ区切りリスト。使用可能な暗号スウィートは、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ウィートを選択します。(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。

5.2.3 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
```

5.2.4 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。

5.3 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。

5.3.1 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);
```

各項目の内容は次のとおりです。

- 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。

5.3.2 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 ― 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 ― 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 ― 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。

5.3.3 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。

6

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)。

7

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. TLS サーバとして機能していて、スタジオからの接続を受け入れるインスタンスで、以下を実行します。
   a. `%SuperServer` TLS 接続を設定します。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. スーパーサーバ・クライアントの TLS 接続を有効にします（スタジオはスーパーサーバ・クライアントであるため）。

      具体的には、**[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 **[スーパーサーバSSL/TLSサポート]** フィールドで、**[有効]** を選択します。
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。

8

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。

8.1 プロセスの概要

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。

8.2 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

8.2.1 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。

8.2.2 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。

8.2.3 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ウィート。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、"暗号スウィート構文の有効化" 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 **[有効な暗号化スウィート]** 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、"暗号スウィート構文の有効化" 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 − 1
- RSA − 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。

8.3 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
```

8.4 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。

9

ミラーリングで TLS を使用するための InterSystems IRIS の構成

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。

9.1 ミラーリングおよび TLS について

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。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) が含まれている必要があります。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。

9.2 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。

9.2.1 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。

9.2.2 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。

9.2.3 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。

10

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

10.1 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

10.1.1 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。

10.1.1.1 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")
   ```

   各項目の内容は次のとおりです。

   - 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。

10.1.2 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。

10.1.2.1 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")
```

   各項目の内容は次のとおりです。

   - 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。

10.2 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。

10.2.1 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。

10.2.1.1 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。

10.2.2 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。

10.2.2.1 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")
```

各項目の内容は次のとおりです。

- 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。

11

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 サーバに関連付けられた `%SuperServer` 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. **[スーパーサーバSSL/TLSサポート]** を選択するには、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 **[有効]** または **[必須]** を選択します。この設定の詳細は、"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。
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 秘密鍵のパスワード]** でパスワードを指定する必要があります。

   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。

12

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ウィ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 12-1: 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。