SYBASE®

設定ガイド

Open Client™/Open Server™

15.5

[Microsoft Windows 版]

ドキュメント ID：DC35837-01-1550-01
改訂：2009 年 11 月

マニュアルの注文

マニュアルの注文を承ります。ご希望の方は、サイベース株式会社営業部または代理店までご連絡ください。マニュアルの変更は、弊社の定期的なソフトウェア・リリース時にのみ提供されます。

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

目次

はじめに

『Open Client/Open Server 設定ガイド Microsoft Windows 版』では、Open Client™/Open Server™ 製品を実行できるようにシステムを設定する方法について説明します。

対象読者

このマニュアルは、システム管理者を対象としています。ここでは、アプリケーションのプログラミングではなく、システム管理の観点から、設定作業とトピックについて説明します。

このマニュアルの内容

このマニュアルには、以下の章があります。

- 「第 1 章 設定の概要」では、設定プロセスの概要と設定に必要な条件について説明します。
- 「第 2 章 Open Client の基本設定」では、クライアント・アプリケーションをサーバに接続する方法について説明し、接続に必要な設定作業を列挙します。
- 「第 3 章 Open Server の基本設定」では、Open Server アプリケーションがクライアント接続要求を受信するための方法について説明し、接続に必要な設定作業を列挙します。
- 「第 4 章 Sybase フェールオーバのための Open Client の設定」では、フェールオーバ中に Open Client アプリケーションがセカンダリ・コンパニオンに接続できるようにするための設定に必要な手順について説明します。
- 「第 5 章 ディレクトリ・サービスの使い方」では、アプリケーションがディレクトリ・サービスから設定情報を取得する方法について説明し、アプリケーションがディレクトリ・サービスを使用するのに必要な設定作業を列挙します。
- 「第 6 章 セキュリティ・サービスの使い方」では、アプリケーションがネットワークをベースとしたセキュリティ・サービスを使用する方法について説明し、アプリケーションがセキュリティ・サービスを使用するのに必要な設定作業を列挙します。また、この章では、LAN Manager と Kerberos の各セキュリティ・サービスに関する情報も提供します。
- 「第 7 章 ocscfg の使用」では、ocscfg ユーティリティを使用して環境変数やドライバを設定する方法について説明します。
- 「第 8 章 dsedit の使用」では、dsedit ユーティリティを使用してディレクトリ・サービスや *sql.ini* ファイルを設定する方法について説明します。

- 「第 9 章 dsedit のトラブルシューティング」では、ocscfg ユーティリティを使用してネットワーク接続を検査する方法について説明します。
- 「付録 A 環境変数」では、Open Client/Open Server 製品で使用する環境変数をリストし、環境変数の設定方法について説明します。
- 「付録 B 設定ファイル」では、設定ファイルの概要を示し、次のファイルについて説明します。
  - *libtcl.cfg* (ドライバ設定ファイル)
  - *sql.ini* ファイル
  - *ocs.cfg* (ランタイム設定ファイル)
- 「付録 C ローカライゼーション」では、ローカライゼーション・ファイルの概要を示し、次のファイルについて説明します。
  - *locales.dat* ファイル
  - *objectid.dat* ファイル
  - ローカライズされたメッセージ・ファイル
  - 照合順ファイル
- 「付録 D Open Client/Open Server の SSL (Secure Socket Layer)」では、Open Client および Open Server の SSL (Secure Sockets Layer) サポートについて説明し、SSL プロトコルを使用するために必要なシステム設定作業について簡単に説明します。

関連マニュアル

詳細については、次のマニュアルを参照できます。

- 『Open Server および SDK 新機能』(Windows 版、Linux 版、UNIX 版、Mac OS X 版) では、Open Server と Software Developer's Kit の新機能について説明しています。このマニュアルは、新機能の提供に伴って改訂されます。
- 使用しているプラットフォームの『Software Developer's Kit リリース・ノート』および『Open Server リリース・ノート』には、リリースに関する最新情報が記載されています。
- 使用しているプラットフォームの『Software Developer's Kit/Open Server インストール・ガイド』では、Open Client/Open Server ソフトウェアのインストール手順について説明しています。
- 『Open Client Client-Library/C リファレンス・マニュアル』には、Open Client Client-Library のリファレンス情報が記載されています。
- 『Open Client DB-Library/C リファレンス・マニュアル』には、DB-Library™ のリファレンス情報が記載されています。
- 『Open Client Client-Library/C プログラマーズ・ガイド』では、Client-Library プログラムの設計および実装方法について説明しています。

- 『Open Server Server-Library/C リファレンス・マニュアル』には、Open Server Server-Library のリファレンス情報が記載されています。
- 『Open Client および Open Server Common Libraries リファレンス・マニュアル』には、CS-Library のリファレンス情報が記載されています。CS-Library は、Client-Library と Server-Library の両方のアプリケーションで役立つユーティリティ・ルーチンの集まりです。
- 『Open Client および Open Server プログラマーズ・ガイド補足』には、Open Client/Open Server 製品を使用するプログラマ向けにプラットフォーム固有の情報が記載されています。このマニュアルには、次の情報が含まれています。
  - アプリケーションのコンパイルおよびリンク
  - Open Client と Open Server 製品に含まれているオンラインのサンプル・プログラム
  - プラットフォーム固有の動作をするルーチン
- 『ASE リファレンス・マニュアル』では、Sybase® Adaptive Server® Enterprise のコマンド、データ型、関数、システム・プロシージャについて説明しています。
- Adaptive Server Enterprise の『Transact-SQL ユーザーズ・ガイド』では、リレーショナル・データベース言語の Sybase の拡張版である Transact-SQL® について説明しています。このマニュアルは、データベース管理システムの操作に慣れていない方のためのテキストとして役立ちます。

その他の情報

Sybase Getting Started CD、SyBooks™ CD、Sybase Product Manuals Web サイトを利用すると、製品について詳しく知ることができます。

- Getting Started CD には、PDF 形式のリリース・ノートとインストール・ガイド、SyBooks CD に含まれていないその他のマニュアルや更新情報が収録されています。この CD は製品のソフトウェアに同梱されています。Getting Started CD に収録されているマニュアルを参照または印刷するには、Adobe Acrobat Reader が必要です (CD 内のリンクを使用して Adobe の Web サイトから無料でダウンロードできます)。
- SyBooks CD には製品マニュアルが収録されています。この CD は製品のソフトウェアに同梱されています。Eclipse ベースの SyBooks ブラウザを使用すれば、使いやすい HTML 形式のマニュアルにアクセスできます。

  一部のマニュアルは PDF 形式で提供されています。これらのマニュアルは SyBooks CD の PDF ディレクトリに収録されています。PDF ファイルを開いたり印刷したりするには、Adobe Acrobat Reader が必要です。

  SyBooks をインストールして起動するまでの手順については、Getting Started CD の『SyBooks インストール・ガイド』、または SyBooks CD の *README.txt* ファイルを参照してください。

- Sybase Product Manuals Web サイトは、SyBooks CD のオンライン版であり、標準の Web ブラウザを使用してアクセスできます。また、製品マニュアルのほか、EBFs/Updates、Technical Documents、Case Management、Solved Cases、ニュース・グループ、Sybase Developer Network へのリンクもあります。

  Sybase Product Manuals Web サイトは、Product Manuals にあります。(http://www.sybase.com/support/manuals/)

Web 上の Sybase 製品の動作確認情報

Sybase Web サイトの技術的な資料は頻繁に更新されます。

❖ **製品認定の最新情報にアクセスする**

1. Web ブラウザで Technical Documents を指定します。(http://www.sybase.com/support/techdocs/)
2. [Partner Certification Report] をクリックします。
3. [Partner Certification Report] フィルタで製品、プラットフォーム、時間枠を指定して [Go] をクリックします。
4. [Partner Certification Report] のタイトルをクリックして、レポートを表示します。

❖ **コンポーネント認定の最新情報にアクセスする**

1. Web ブラウザで Availability and Certification Reports を指定します。(http://certification.sybase.com/)
2. [Search By Base Product] で製品ファミリとベース製品を選択するか、[Search by Platform] でプラットフォームとベース製品を選択します。
3. [Search] をクリックして、入手状況と認定レポートを表示します。

❖ **Sybase Web サイト (サポート・ページを含む) の自分専用のビューを作成する**

MySybase プロファイルを設定します。MySybase は無料サービスです。このサービスを使用すると、Sybase Web ページの表示方法を自分専用にカスタマイズできます。

1. Web ブラウザで Technical Documents を指定します。(http://www.sybase.com/support/techdocs/)
2. [MySybase] をクリックし、MySybase プロファイルを作成します。

Sybase EBF とソフトウェア・メンテナンス

❖ EBF とソフトウェア・メンテナンスの最新情報にアクセスする

1 Web ブラウザで Sybase Support ページを指定します。(http://www.sybase.com/support)

2 [EBFs/Maintenance] を選択します。MySybase のユーザ名とパスワードを入力します。

3 製品を選択します。

4 時間枠を指定して [Go] をクリックします。EBF/Maintenance リリースの一覧が表示されます。

鍵のアイコンは、「Technical Support Contact」として登録されていないため、一部の EBF/Maintenance リリースをダウンロードする権限がないことを示しています。未登録でも、Sybase 担当者またはサポート・コンタクトから有効な情報を得ている場合は、[Edit Roles] をクリックして、「Technical Support Contact」の役割を MySybase プロファイルに追加します。

5 EBF/Maintenance レポートを表示するには [Info] アイコンをクリックします。ソフトウェアをダウンロードするには製品の説明をクリックします。

表記規則

表 *1:* 構文の表記規則

キー	定義
command	コマンド名、コマンドのオプション名、ユーティリティ名、ユーティリティのフラグ、キーワードは sans serif で示す。
variable	変数 (ユーザが入力する値を表す語) は斜体で表記する。
{ }	中カッコは、その中から必ず 1 つ以上のオプションを選択しなければならないことを意味する。コマンドには中カッコは入力しない。
[ ]	角カッコは、オプションを選択しても省略してもよいことを意味する。コマンドには角カッコは入力しない。
( )	このカッコはコマンドの一部として入力する。
|	中カッコまたは角カッコの中の縦線で区切られたオプションのうち 1 つだけを選択できることを意味する。
,	中カッコまたは角カッコの中のカンマで区切られたオプションをいくつでも選択できることを意味する。複数のオプションを選択する場合には、オプションをカンマで区切る。

アクセシビリティ機能

このマニュアルには、アクセシビリティを重視した HTML 版もあります。この HTML 版マニュアルは、スクリーン・リーダーで読み上げる、または画面を拡大表示するなどの方法により、その内容を理解できるよう配慮されています。

Open Client および Open Server のマニュアルは、連邦リハビリテーション法第 508 条のアクセシビリティ規定に準拠していることがテストにより確認されています。第 508 条に準拠しているマニュアルは通常、World Wide Web Consortium (W3C) の Web サイト用ガイドラインなど、米国以外のアクセシビリティ・ガイドラインにも準拠しています。

注意 アクセシビリティ・ツールを効率的に使用するには、設定が必要な場合もあります。一部のスクリーン・リーダーは、テキストの大文字と小文字を区別して発音します。たとえば、すべて大文字のテキスト (ALL UPPERCASE TEXT など) はイニシャルで発音し、大文字と小文字の混在したテキスト (Mixed Case Text など) は単語として発音します。構文規則を発音するようにツールを設定すると便利かもしれません。詳細については、ツールのマニュアルを参照してください。

Sybase のアクセシビリティに対する取り組みについては、Sybase Accessibility (http://www.sybase.com/accessibility) を参照してください。Sybase Accessibility サイトには、第 508 条と W3C 標準に関する情報へのリンクもあります。

不明な点があるときは

Sybase ソフトウェアがインストールされているサイトには、Sybase 製品の保守契約を結んでいるサポート・センタとの連絡担当の方 (コンタクト・パーソン) を決めてあります。マニュアルだけでは解決できない問題があった場合には、担当の方を通して Sybase のサポート・センタまでご連絡ください。

第 1 章

設定の概要

このマニュアルは、Microsoft Windows 版 Open Client/Open Server の設定ガイドです。『Software Developer's Kit および Open Server インストール・ガイド Microsoft Windows 版』の指示に従って、Open Client または Open Server をインストールしてから、このマニュアルを読んでください。Open Client は、SDK (Software Developer's Kit) の一部です。

この章では、Open Client と Open Server の設定プロセスの概要について説明します。

Open Client と Open Server について

Open Client は、アプリケーション・プログラミング・インタフェース (API: Application Programming Interface) と Net-Library を提供します。これらを使用することで、Adaptive Server Enterprise と、Open Server アプリケーション、カスタマ・アプリケーション、サード・パーティ製品、その他の Sybase 製品との間で通信することが可能になります。

Open Server は、カスタム・サーバの作成に必要なツールとインタフェースを提供します。Open Client と同様に、プログラミング API と Net-Library がクライアントや他のサーバとの通信を可能にします。さらに Open Server は、次の機能を持つルーチンも提供します。

- 複数のクライアント接続を処理するルーチン
- クライアントとの対話をスケジュールするルーチン
- エラー条件を処理するルーチン
- サーバから要求されたその他の機能を実行する。

Open Client と Open Server の詳細については、次のマニュアルを参照してください。

- 『Open Client Client-Library/C リファレンス・マニュアル』
- 『Open Server Server-Library/C リファレンス・マニュアル』
- 『Open Client DB-Library/C リファレンス・マニュアル』

設定の概要

Open Client/Open Server ソフトウェアを正しく機能させるには、特定の情報が必要です。設定とは、この情報を使用できるようにシステムを準備するプロセスです。

Open Client と Open Server は、設定された情報を使用して次の処理を行います。

- Open Client または Open Server アプリケーションを初期化する。
- Adaptive Server Enterprise または Open Server アプリケーションとの接続を確立する。

注意 アプリケーションが最新機能に確実にアクセスできるようにするには、バージョンを CS_CURRENT_VERSION に設定します。

初期化プロセス

❖ **Open Client/Open Server アプリケーションの初期化**

1 SYBASE 環境変数を使用して Sybase インストール・ディレクトリのロケーションを決定します。

2 ロケール固有 POSIX 環境変数 LC_*、LANG、LC_ALL、LC_COLLATE、*locales.dat* ファイルを使用して、アプリケーションがどの言語、文字セット、照合順を使用するかを決定します。

3 *libtcl.cfg* ファイルを使用して、必要に応じてディレクトリ・ドライバとセキュリティ・ドライバをロードします。

接続プロセス

クライアントとサーバは接続を介して通信します。クライアント・アプリケーションがサーバ・アプリケーションに接続するには、サーバ・アプリケーションがクライアントの接続要求を受信している必要があります。

❖ **Open Client から接続を確立する**

1 DSQUERY 環境変数を、ターゲット・サーバの名前の決定に使用します。

2 *sql.ini* ファイルまたはディレクトリ・サービスを使用して、ターゲット・サーバのアドレスを取得します。

注意 Open Client が DSQUERY を使用するのは、Open Client アプリケーションがサーバ名を指定していない場合だけです。

❖ Open Server で要求を受信する

1　DSLISTEN 環境変数を使用して Open Server アプリケーションの名前を決定します。

2　*sql.ini* ファイルまたはディレクトリ・サービスを使用して、Open Server アプリケーションのアドレスを取得します。

注意　DSLISTEN を使用するのは、Open Server アプリケーションが初期化時にサーバを指定していない場合だけです。

設定作業

Open Client/Open Server 製品がアプリケーションを初期化して接続を行う前に、いくつかの基本的な設定作業を行う必要があります。

次のような設定作業があります。

- ターゲットのデフォルト・サーバと初期ローカライゼーション値を指定するように、環境変数を設定します。Open Client アプリケーションと Open Server アプリケーションがサーバ名を指定していない場合は、DSQUERY と DSLISTEN の値が使用されます。
- ターゲット・サーバのアドレスが使用可能かどうかを確認します。
- 必要であれば、ネットワーク・ドライバを設定します。

次のような場合には、追加作業が必要です。

- ディレクトリ・サービスを使用する。
- セキュリティ・サービスを使用する。
- 初期ローカライゼーション値のほかにカスタム・ローカライゼーション値を使用したり、初期ローカライゼーション値の代わりにカスタム・ローカライゼーション値を使用する。

第 2 章

Open Client の基本設定

この章では、Open Client に必要な基本設定について説明します。

注意 注意書きがある場合を除いて、この章の内容は DB-Library と Client-Library の両方に適用されます。特に DB-Library は初期ローカライゼーション値を決定するのに環境変数を使用せず、*libtcl.cfg* ファイルを調べません。ただし、SYBASE 環境変数と DSQUERY 環境変数は調べます。DB-Library の詳細については、『Open Client DB-Library/C リファレンス・マニュアル』を参照してください。

基本設定の概要

すべての Open Client アプリケーションは、次のような基本設定情報を必要とします。これらの情報は、初期化時と接続時に取得されます。

- Sybase インストール・ディレクトリのロケーション
- ロケール名
- ローカライズされたメッセージ・ファイルと文字セット・ファイル
- ターゲット・サーバ名
- ターゲット・サーバのネットワーク・アドレス
- 使用するセキュリティ・メカニズム

SYBASE 環境変数によって定義される Sybase インストール・ディレクトリのロケーション

バージョン 10.x 以降に構築されたアプリケーションを使用する異機種間環境で、Open Client と Open Server の現在のバージョンのインストール先のロケーションを指すには、コマンド・プロンプトで SYBASE 環境変数、SYBASE_OCS 環境変数、PATH 環境変数を明示的に設定する必要があります。

たとえば、*C:¥SYBASE* にインストールされているバージョン 15.5 製品を使用するアプリケーションの場合は、コマンド・プロンプトを開き、次の環境変数を設定します。

```
set SYBASE=C:¥SYBASE
set SYBASE_OCS=OCS-15_0
set PATH=%PATH%;%SYBASE%¥%SYBASE_OCS%¥bin;
%SYBASE%¥%SYBASE_OCS%¥dll
```

ロケール名

Open Client は次の POSIX 環境変数の値をロケール名として使用します (DB-Library には適用されません)。

- LC_ALL
- LANG (LC_ALL が定義されていない場合)

Open Client はあとからこの値を使用して *locales.dat* ファイルからローカライゼーション情報を取得します。LC_ALL、LANG、Language が定義されていない場合、Open Client はロケール名として “default” を使用します。

ローカライズされたメッセージ・ファイルと文字セット・ファイル

Open Client は、*locales.dat* ファイルを調べて、上記の手順で指定したロケール名と一致するエントリを探し、*locales.dat* ファイルに設定されているローカライズされたメッセージ・ファイルと文字セット・ファイルをロードします。

ターゲット・サーバの名前

Open Client は、いずれかのソースから次の順序でターゲット・サーバの名前を取得します。

1 ct_connect (または dbopen) に対する呼び出しにサーバ名を指定できるクライアント・アプリケーション。

2 アプリケーションにターゲット・サーバが指定されていない場合は、DSQUERY 環境変数。

3 DSQUERY が設定されていない場合は、デフォルト名の SYBASE。

ターゲット・サーバのネットワーク・アドレス

Open Client は、ディレクトリ・サービスまたは *sql.ini* ファイルからターゲット・サーバのアドレスを取得します。

- ディレクトリ・サービス − Open Client は、*libtcl.cfg* ファイルの [NT_DIRECTORY] セクション内のエントリを探して、サーバ・アドレス情報の検索先を決定します。CS_DS_PROVIDER プロパティの設定値によって、アプリケーションがどの [NT_DIRECTORY] エントリを検索するかが決定されます。プロパティが設定されていない場合は、[NT_DIRECTORY] セクションの最初のエントリがデフォルトで使用されます。

- *sql.ini* ファイル – ディレクトリ・サービスが使用されていない場合、または使用されていても機能していない場合は、Open Client は *sql.ini* ファイルを調べて、名前と一致する SERVERNAME エントリを検出し、対応するターゲット・アドレスを使用します。

  sql.ini ファイルの詳細については、「sql.ini ファイル」(69 ページ) を参照してください。

  バージョン 10.x 以降用に構築されたアプリケーションを使用する異機種間環境でも、アドレス・ファイル名を各アプリケーションに渡すことによって、単一の *sql.ini* ファイルを管理できます。次に例を示します。

  ```
  isql -Usa -P -Sconnect50 -Ic:¥sybase¥ini¥sql.ini
  ```

使用するセキュリティ・メカニズム

(DB-Library には適用されません) クライアント・アプリケーションがネットワークベースのセキュリティ・サービスを要求している場合は、*libtcl.cfg* ファイルの [SECURITY] セクションを調べて、使用するセキュリティ・ドライバを決定します。

設定作業

クライアント・アプリケーションが前述の処理を実行できるようにするには、以下の項の作業を行います。

環境変数の設定

❖ **環境変数を設定する**

1. LC_ALL または LANG 環境変数を任意のロケール名に設定します。指定するロケール名は、*locales.dat* ファイルのエントリに対応している必要があります。LC_ALL または LANG を設定しない場合は、*locales.dat* の “default” エントリに、アプリケーションで使用するローカライゼーション値が反映されていることを確認します。

   ロケール・ファイルに指定されている言語、文字セット、照合順と一致するローカライゼーション・ファイルがあることを確認してください。

2. アプリケーションがカスタム・ローカライゼーション値を使用する場合は、LC_ALL、LC_COLLATE、LC_TYPE、LC_MESSAGE、または LC_TIME 環境変数をロケール名に設定します。

   アプリケーションがどの環境変数を使用するかわからない場合は、すべての環境変数を希望のロケール名に設定してください。

3 DSQUERY 環境変数をターゲット・サーバの名前に設定します。クライアント・アプリケーションにターゲット・サーバの名前が指定されている場合、DSQUERY を設定する必要はありません。DSQUERY が設定されておらず、アプリケーションにもサーバ名が指定されていない場合には、Open Client はサーバ名として "SYBASE" を使用します。

ocscfg ユーティリティを使用して環境変数を設定する方法については、「環境変数の設定」(42 ページ) を参照してください。

ドライバの設定

ディレクトリとセキュリティ・ドライバを設定するには、ocscfg ユーティリティを使用します。

ドライバの設定方法については、「第 7 章 ocscfg の使用」を参照してください。

ドライバと *libtcl.cfg* ファイルの詳細については、「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照してください。

sql.ini ファイルの設定

❖ **sql.ini ファイルを設定する**

1 dsedit ユーティリティを使用して *sql.ini* ファイルにターゲット・サーバのエントリを作成します。

2 SERVERNAME 要素が DSQUERY 環境変数の値と一致するエントリが *sql.ini* ファイルにあることを確認します。

sql.ini ファイルに情報を追加する方法については、「第 8 章 dsedit の使用」を参照してください。*sql.ini* ファイルの詳細については、「sql.ini ファイル」(69 ページ) を参照してください。

第 3 章

Open Server の基本設定

この章では、Open Server に必要な基本設定について説明します。

Open Server アプリケーションについて

Open Server アプリケーションは、機能的に次の 3 つのタイプに分けられます。

- スタンドアロン
- 補助
- ゲートウェイ

Open Server アプリケーションの設定はタイプによって異なります。Open Server アプリケーションのタイプの詳細については、『Open Server Server-Library/C リファレンス・マニュアル』を参照してください。

基本設定の概要

すべての Open Server アプリケーションは、次のような基本設定情報を必要とします。これらの情報は、初期化時と接続時に取得されます。

- Sybase インストール・ディレクトリのロケーション
- ロケール名
- ローカライズされたメッセージ・ファイルと文字セット・ファイル
- ターゲット・サーバの名前
- ターゲット・サーバのネットワーク・アドレス

SYBASE 環境変数によって定義される Sybase インストール・ディレクトリのロケーション

バージョン 10.x 以降に構築されたアプリケーションを使用する異機種間環境で、Open Client と Open Server の現在のバージョンのインストール先のロケーションを指すには、コマンド・プロンプトで SYBASE 環境変数、SYBASE_OCS 環境変数、PATH 環境変数を明示的に設定する必要があります。

たとえば、*C:¥SYBASE* にインストールされている 15.5 製品を使用するアプリケーションの場合は、コマンド・プロンプトを開き、次の環境変数を設定します。

```
set SYBASE=C:¥SYBASE
set SYBASE_OCS=OCS-15_0
set PATH=%PATH%;%SYBASE%¥%SYBASE_OCS%¥bin;
%SYBASE%¥%SYBASE_OCS%¥dll
```

ロケール名

Open Server は次の POSIX 環境変数の値をロケール名として使用します。

- LC_ALL
- LANG (LC_ALL が定義されていない場合)

Open Server はあとでこの値を使用して *locales.dat* ファイルからローカライゼーション情報を取得します。環境変数が定義されていない場合、Open Server はロケール名として “default” を使用します。

ローカライズされたメッセージ・ファイルと文字セット・ファイル

Open Server は *locales.dat* ファイルを調べて、名前が上記の手順 2 で指定したロケール名と一致するエントリを探し、*locales.dat* ファイルに指定されているローカライズされたメッセージ・ファイルと文字セット・ファイルをロードします。

ターゲット・サーバの名前

ターゲット・サーバの名前。Open Server は、次のソースのいずれかからこの順序で Open Server アプリケーションの名前を取得します。

1 srv_init に対する呼び出しにサーバ名を指定できる Open Server アプリケーション。

2 アプリケーションにターゲット・サーバ名が指定されていない場合、DSLISTEN 環境変数。

3 DSLISTEN が設定されていない場合は、デフォルト名の SYBASE。

ターゲット・サーバのネットワーク・アドレス

ターゲット・サーバのネットワーク・アドレス。Open Server は、ディレクトリ・サービス、または *sql.ini* からターゲット・サーバのアドレスを取得します。

- ディレクトリ・サービス − Open Server は *libtcl.cfg* ファイルの [NT_DIRECTORY] セクション内のエントリを探して、サーバ・アドレス情報の検索先を決定します。CS_DS_PROVIDER プロパティの設定値によって、アプリケーションがどの [NT_DIRECTORY] エントリを検索するかが決定されます。プロパティが設定されていない場合は、[NT_DIRECTORY] セクションの最初のエントリがデフォルトで使用されます。

- *sql.ini* ファイル – ディレクトリ・サービスが使用されていない場合、または使用されていても機能していない場合は、Open Server は *sql.ini* ファイルを調べて、上記の手順 4 で指定した名前と一致する SERVERNAME エントリを検出し、対応するターゲット・アドレスを使用します。

  リリース 10.0.x 以降用に構築されたアプリケーションを使用する異機種間環境では、アドレス・ファイル名を明示的に各アプリケーションに渡すことによって、単一の *sql.ini* ファイルを管理できます。次に例を示します。

  ```
  isql -Usa -P -Sconnect50 -Ic:¥sybase¥ini¥sql.ini
  ```

ネットワークベースのセキュリティ・メカニズムを使用する接続をクライアントが要求している場合、Open Server は *libtcl.cfg* の [SECURITY] セクションで該当するセキュリティ・ドライバを探します。

設定作業

Open Server アプリケーションが前述の処理を実行できるようにするには、次の作業を行います。

- sql.ini またはレジストリの設定
- 環境変数の設定
- ドライバの設定

それぞれの作業について、以下の項で説明します。

sql.ini またはレジストリの設定

❖ *sql.ini* ファイルを設定する

1. dsedit を使用して、*sql.ini* にサーバの名前とディレクトリのエントリを作成します。
2. SERVERNAME 要素が DSLISTEN 環境変数の値と一致するエントリが *sql.ini* ファイルにあることを確認します。

❖ レジストリを設定する

1. [コントロール パネル] の [システム] を開きます。
2. [詳細設定] タブの [環境変数] をクリックします。

3 次のシステム変数を編集します。

a DSLISTEN − *sql.ini* またはディレクトリ・サービスにリストされている Open Server アプリケーションの名前に値を設定します。

b DSQUERY − *sql.ini* またはディレクトリ・サービスにリストされているターゲット・サーバの名前に値を設定します。

4 環境変数ウィンドウで [OK] をクリックして、新しい値を設定します。

dsedit の使用方法については、「第 8 章 dsedit の使用」を参照してください。

sql.ini の詳細については、「sql.ini ファイル」(69 ページ) を参照してください。

環境変数の設定

次の環境変数を設定します。

- LC_ALL または LANG 環境変数を任意のロケール名に設定します。

  指定するロケール名は、*locales.dat* ファイルのエントリに対応している必要があります。LC_ALL または LANG を設定しない場合は、*locales.dat* の “default” エントリに、アプリケーションで使用するローカライゼーション値が反映されていることを確認します。

  ロケール・ファイルに指定されている言語、文字セット、照合順と一致するローカライゼーション・ファイルがあることを確認してください。

- アプリケーションが「カスタム・ローカライゼーション値」を使用する場合は、LC_ALL、LC_COLLATE、LC_TYPE、LC_MESSAGE、または LC_TIME 環境変数をロケール名に設定します。

  アプリケーションがどの環境変数を使用するかわからない場合は、すべての環境変数を希望のロケール名に設定してください。

- DSLISTEN 環境変数を Open Server アプリケーションの名前に設定します。

  アプリケーションに Open Server アプリケーションの名前が指定されている場合、DSLISTEN を設定する必要はありません。DSLISTEN が設定されていなくて、アプリケーションにもサーバ名が指定されていない場合には、Open Server はサーバ名として SYBASE を使用します。

- Open Server アプリケーションがゲートウェイ・アプリケーションとして機能する場合、DSQUERY 環境変数はターゲット・サーバの名前に設定してください。

ocscfg ユーティリティを使用して環境変数を設定する方法については、「環境変数の設定」(42 ページ) を参照してください。

ドライバの設定

ocscfg ユーティリティを使用して、ネットワーク、ディレクトリ、セキュリティ・ドライバを設定します。

ドライバの設定方法については、「第 7 章 ocscfg の使用」を参照してください。

ドライバと *libtcl.cfg* ファイルの詳細については、「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照してください。

第 4 章

Sybase フェールオーバのための Open Client の設定

Sybase のフェールオーバ機能については、Adaptive Server Enterprise の『高可用性システムにおける Sybase フェールオーバの使用』ガイドに記載されています。この章では、フェールオーバの間にセカンダリ・コンパニオンに接続するよう Open Client アプリケーションを設定する場合に必要な手順について説明します。この情報は、上記のマニュアルには含まれていません。

注意 DB-Library は動的 HA フェールオーバをサポートしません。C 版および COBOL 版の Embedded SQL™ (ESQL) は、バージョン 12.5 以降、HA フェールオーバをサポートしています。

sql.ini ファイルへの hafailover 行の追加

プライマリ・コンパニオンがクラッシュしたり、shutdown または shutdown with nowait を発行して、フェールオーバが発生した場合は、フェールオーバ・プロパティのあるクライアントは、セカンダリ・コンパニオンに自動的に再接続します。クライアントにフェールオーバ・プロパティを付与するには、*sql.ini* ファイルに “hafailover” という行を追加して、クライアントがセカンダリ・コンパニオンに接続するために必要な情報を提供する必要があります。この行を追加するには、ファイル・エディタか dsedit ユーティリティを使用します。

次に、“MONEY1” コンパニオンと “PERSONNEL1” コンパニオンを対称型に設定する場合の *sql.ini* のエントリを示します。

```
[MONEY1]
 master=TCP,FN1,9835
 query=TCP,FN1,9835
 hafailover=PERSONNEL1


[PERSONNEL1]
 master=TCP,HUM1,7586
 query=TCP,HUM1,7586
 hafailover=MONEY1
```

sql.ini ファイルにこの情報を追加する方法の詳細については、「sql.ini ファイルの設定」(8 ページ) を参照してください。

注意 クライアント・アプリケーションは、フェールオーバによって送信できなかったクエリを再送する必要があります。また、カーソル宣言などの接続固有のその他の情報をリストアする必要があります。

Client-Library アプリケーションの変更

注意 クラスタにインストールされているアプリケーションは、プライマリ・コンパニオンとセカンダリ・コンパニオンの両方で実行可能でなければなりません。並列設定が必要なアプリケーションをインストールする場合は、フェールオーバの間にセカンダリ・コンパニオンがアプリケーションを実行できるように、セカンダリ・コンパニオンにも並列処理の設定を行う必要があります。

Client-Library 呼び出しで記述されたアプリケーションを、Sybase フェールオーバ・ソフトウェアで実行できるようにするには、変更が必要です。変更するには、次の手順に従います。

1 ct_config および ct_con_props の各 Client-Library API 呼び出しを使用して、CS_HAFAILOVER プロパティを設定します。プロパティの有効値は CS_TRUE と CS_FALSE です。デフォルト値は CS_FALSE です。このプロパティは、コンテキスト・レベルと接続レベルのどちらでも設定することができます。次に、コンテキスト・レベルでプロパティを設定する例を示します。

```
CS_BOOL bhafailover = CS_TRUE;
retcode = ct_config(context, CS_SET, CS_HAFAILOVER,
&bhafailover, CS_UNUSED, NULL);
```

次に、接続レベルでのプロパティ設定を示します。

```
CS_BOOL bhafailover = CS_FALSE;
retcode = ct_con_props(connection, CS_SET,
CS_HAFAILOVER, &bhafailover, CS_UNUSED, NULL);
```

2　フェールオーバ・メッセージを処理します。コンパニオンがシャットダウン処理を始めると、クライアントはフェールオーバが発生するという情報メッセージを受け取ります。これは、クライアント・エラー・ハンドラの情報メッセージとして扱ってください。

3　フェールオーバ設定を確認します。フェールオーバ・プロパティを設定し、*sql.ini* ファイルにセカンダリ・コンパニオン・サーバの有効なエントリが設定されていると、接続はフェールオーバ接続になり、クライアントは適切に再接続します。

ただし、フェールオーバ・プロパティが設定されていても、*sql.ini* ファイルに hafailover サーバのエントリがない場合 (またはその逆の場合) は、フェールオーバ接続にはなりません。この場合は、フェールオーバ・プロパティがオフになった、高可用性ではない通常の接続になります。フェールオーバ・プロパティを確認して、接続がフェールオーバ接続かどうかを確認してください。これを行うには、CS_GET の *action* とともに ct_con_props を呼び出します。

4　リターン・コードを検証します。フェールオーバが成功したら、ct_results と ct_send を呼び出すと、接続のタイプに従って CS_RET_HAFAILOVER が返されます。

- 同期接続では、API 呼び出しは CS_RET_HAFAILOVER を直接返します。
- 非同期接続では、API は CS_PENDING を返し、コールバック機能は CS_RET_HAFAILOVER を返します。

リターン・コードによっては、next コマンドを送信して実行するなど、アプリケーションは必要なプロセスを行います。

5　オプション値をリストアします。クライアントがプライマリ・コンパニオンから切断されると、このクライアント接続に合わせて設定してある set オプション (たとえば、set role など) は失われます。フェールオーバした接続で、これらのオプションをリセットします。

6　アプリケーションを再構築し、フェールオーバ・ソフトウェアに含まれるライブラリにリンクさせます。

注意　sp_companion resume を発行するまで、フェールオーバ・プロパティ (isql -Q など) が設定されたクライアントを接続できません。sp_companion prepare_failback を発行してからクライアントを再接続しようとすると、クライアントは sp_companion resume を発行するまでハングします。

Sybase フェールオーバでの isql の使い方

isql を使用してフェールオーバ機能のあるプライマリ・サーバに接続するには、次の手順に従います。

- *sql.ini* ファイル・エントリで指定されているセカンダリ・コンパニオン・サーバのあるプライマリ・サーバを選択します。
- -Q コマンド・ライン・オプションを使用します。

sql.ini ファイルに、「sql.ini ファイルへの hafailover 行の追加」に示されているエントリ例がある場合は、次のように入力して、フェールオーバで isql を使用できます。

```
isql -S PERSONNEL1 -Q
```

第 5 章

ディレクトリ・サービスの使い方

Client-Library と Server-Library アプリケーションはディレクトリ・サービスを使用して、サーバに関する情報を記録します。この章では、ディレクトリ・サービスの機能と設定方法について説明します。

注意 DB-Library は LDAP ディレクトリ・サービスのみをサポートしています。

ディレクトリ・サービスの概要

ディレクトリ・サービスは、ネットワーク・エンティティに関する情報の作成、修正、検索を管理します。*sql.ini* ファイルの代わりに、Client-Library と Server-Library アプリケーションはディレクトリ・サービスを使用してサーバについての情報を取得します。

ディレクトリ・サービスを使用する利点は、新しいサーバをネットワークに追加するときやサーバを新しいアドレスに移動するときに、複数の *sql.ini* ファイルを更新する必要がない点です。

使用するディレクトリ・サービス・プロバイダは、プラットフォームによってそれぞれ異なります。Microsoft Windows では、Windows レジストリと LDAP を使用できます。

LDAP

LDAP (Lightweight Directory Access Protocol) は、ディレクトリ・リストへのアクセスに使用します。ディレクトリ・リストやサービスは、ネットワーク上のユーザとリソースの名前、プロファイル情報、マシン・アドレスを提供します。ユーザ・アカウントとネットワーク・パーミッションを管理するのに、これを使用できます。

通常、LDAP サーバは階層構造であり、リソースを高速検索できます。従来の Sybase *sql.ini* ファイルの代わりに、LDAP を使用して Sybase サーバの情報を保管したり検索したりできます。

LDAP サービスは、どのようなタイプでも (実際のサーバであっても、その他の LDAP サービスへのゲートウエイであっても)、LDAP サーバと呼ばれます。LDAP ドライバは LDAP クライアント・ライブラリを呼び出して、LDAP サーバへの接続を確立します。LDAP ドライバとクライアント・ライブラリは、暗号化を有効にするかどうかなどの通信プロトコル、およびクライアントとサーバの間で交換されるメッセージのコンテンツを定義します。メッセージとは、データ・フォーマット情報も含めたクライアントの読み込み、書き込み、クエリ、サーバ応答などの要求です。

LDAP ディレクトリ・サービスと Sybase *sql.ini* ファイルの比較

LDAP ディレクトリ・サービスは、通常の Sybase *sql.ini* ファイルの代わりとなるもので、サーバ情報を「フラット」ファイルに格納しています。そのため、*sql.ini* ファイルのサーバ情報を変更するときは、サイトの全マシン (クライアントとサーバ) を更新する必要があります。

LDAP ディレクトリ・サービスを使用すると、中央レポジトリにユーザ、リソース、セキュリティの情報を統合することで、リソース情報を非常に管理しやすくなります。また、LDAP サービスは次のものを提供します。

- ユーザ、ソフトウェア、リソース、ネットワーク、ファイルなどの情報の単一の階層表示
- サーバと分散エンタープライズ・アプリケーションのシングル・サインオン
- 機密データのアクセス管理のためのユーザ・ログインと役割情報

ユーザ役割は、システム管理者などの個人、または経理部門などのユーザ・グループ全体に割り当てられます。役割によって、ユーザがアクセスできる情報とサーバ、およびユーザの読み込み／書き込みパーミッション (ユーザにこれらのパーミッションがある場合) が決まります。同じユーザ役割を持つ複数ユーザを少数のサーバ接続に多重化して、リソースを節約しスケーラビリティを拡大できます。

表 5-1 に、Sybase *sql.ini* ファイルと LDAP サーバの相違点を示します。

表 *5-1: sql.ini* ファイルと *LDAP* ディレクトリ・サービスの比較

***sql.ini* ファイル**	**ディレクトリ・サービス**
プラットフォーム固有	プラットフォームに依存しない
Sybase インストール環境ごとに異なった構造	統一された階層構造
マスタ・エントリとクエリ・エントリが別々に存在する	クライアントとサーバの両方がアクセスするサーバごとに 1 エントリを含む
サーバのメタデータを格納できない	サーバのメタデータを格納する

TCP 接続とフェールオーバ・マシンを指定する従来の *sql.ini* ファイルは、次のようになります。

```
[MONEY]
master=TCP, huey, 5000
query=TCP, huey, 5000
hafailover=PERSONEL
[PERSONEL]
master=TCP, huey, 5000
query=TCP, huey, 5000
hafailover=MONEY
```

次に、TCP とフェールオーバ・マシンを指定した LDAP エントリの例を示します。

```
dn: sybaseServername=foobar, dc=sybase,dc=com
objectClass: sybaseServer
sybaseVersion: 15501
sybaseServername: foobar
sybaseService: ASE
sybaseStatus: 4
sybaseAddress: TCP#1#foobar 5000
sybaseRetryCount: 12
sybaseRetryDelay: 30
sybaseHAServernam: secondary
```

LDAP ディレクトリ・サービスのすべてのエントリは、エンティティと呼ばれます。各エンティティは DN (識別名) を持ち、それぞれの DN に基づいて階層ツリー構造内に格納されます。このツリーは、ディレクトリ情報ツリー (DIT) と呼ばれます。接続中に DIT ベースを指定することで、クライアント接続は LDAP サーバの検索開始位置を指定します。表 5-2 に、DIT ベースの有効な値を示します。

表 *5-2: Sybase LDAP* エントリ定義

属性名	値のタイプ	説明
sybaseVersion	整数	サーバのバージョン番号。
sybaseServername	文字列	サーバの名前。
sybaseService	文字列	サービスの種類。Sybase Adaptive Server。
sybaseStatus	整数	ステータス：1 = アクティブ、2 = 停止、3 = 失敗、4 = 不明。
sybaseAddress	文字列	アドレス文字列の各エントリは # 文字で区切る。各サーバのアドレスには、次の項目が含まれる。 • プロトコル：TCP、NAMEDPIPE • sybaseStatus の値 • アドレス：そのプロトコル・タイプの有効なアドレス
sybaseSecurity (オプション)	文字列	セキュリティ OID (オブジェクト ID)。
sybaseRetryCount	整数	この属性は、CS_RETRY_COUNT にマップされる。CS_RETRY_COUNT は、ct_connect がサーバ名と対応するネットワーク・アドレスのシーケンスをリトライする回数を指定する。
sybaseRetryDelay	整数	この属性は、CS_LOOP_DELAY にマップされる。CS_LOOP_DELAY は、ct_connect がアドレスのすべてのシーケンスをリトライするまでの遅延時間を秒単位で指定する。
sybaseHAservername (オプション)	文字列	フェールオーバ保護用のセカンダリ・サーバ。

%SYBASE%¥%SYBASE_OCS%¥ini に、次の LDAP サービスの LDAP ディレクトリ・スキーマが用意されています。

- *sybase.schema* － OpenLDAP サーバで使用するディレクトリ・スキーマが格納されている。
- *sybase-schema.conf* － Netscape 固有の構文を使用したディレクトリ・スキーマが格納されている。
- *sybase.ldf* － Microsoft Active Directory 用の Unicode フォーマットのディレクトリ・スキーマが格納されている。

前の例のエンティティは、“foobar” という名前の Adaptive Server がポート番号 5000 の TCP 接続で受信していることを示しています。また、このエンティティでは、12 (回) のリトライ回数と 30 (秒) のリトライ遅延時間も指定しています。sybaseRetryCount と sybaseRetryDelay は、それぞれ CS_RETRY_COUNT と CS_LOOP_DELAY にマップされます。Client-Library はサーバから応答があるアドレスを見つけると、Client-Library とサーバ間でログイン・ダイアログが開始されます。ログインが失敗しても、Client-Library は他のアドレスをリトライすることはありません。

最も重要なエンティティは address 属性です。この属性には、サーバへの接続を設定する方法と、サーバが受信接続を待機する方法についての情報があります。エントリを異なるプラットフォームの異なる Sybase 製品で使用できるようにするには、アドレス属性のプロトコル・フィールドとアドレス・フィールド (たとえば、“TCP” と “foobar 5000”) を、プラットフォームや製品に依存しない形式にする必要があります。

LDAP では各属性の複数のエントリをサポートしているので、各アドレス属性は単一サーバのアドレス (プロトコル、アクセス・タイプ、アドレスを含む) を持つ必要があります。詳細については、表 5-2 (22 ページ) の sybaseAddress を参照してください。

次に、接続プロトコルが異なる 2 つのアドレスで受信している Microsoft Windows サーバの LDAP エントリの例を示します。

```
sybaseAddress = TCP#1#TOEJAM 4444
sybaseAddress = NAMEPIPE#1#¥pipe¥sybase¥query
```

アドレス・フィールドの各エントリは、# 文字で区切ります。アドレス属性の各フィールドの値の定義については、表 5-2 (22 ページ) を参照してください。

サーバ・オブジェクトと属性

ディレクトリ・サービスには、Open Client がアクセスするサーバに関する情報が入っていなければなりません。

ディレクトリ・サービスはサーバ・エントリをディレクトリ・オブジェクトとして識別します。各ディレクトリ・オブジェクトには、ユニークな属性のセットがあります。サーバ・オブジェクト・エントリは、dsedit ユーティリティを使用して作成、表示、修正できます。詳細については、「第 8 章 dsedit の使用」を参照してください。

ディレクトリ・ドライバ

Open Client/Open Server ソフトウェアは、ディレクトリ・ドライバを使用してディレクトリ・サービスから情報を取得します。

ディレクトリ・ドライバは、特定のディレクトリ・サービスに対する汎用インタフェースを Open Client/Open Server ソフトウェアに提供する、動的にリンクされた Sybase ライブラリです。Sybase はサポートするディレクトリ・サービスごとにディレクトリ・ドライバを提供しています。

ディレクトリ・ドライバは、*libtcl.cfg* ファイルにリストされています。ディレクトリ・ドライバと *libtcl.cfg* ファイルの詳細については、「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照してください。

アプリケーションがディレクトリ・サービスを使用する仕組み

Client-Library、Server-Library、DB-Library は、次のようにしてディレクトリ・サービスと *sql.ini* のどちらを使用するかを決定します。

1 Client-Library または Server-Library アプリケーションがディレクトリ・ドライバを指定している場合 (Client-Library では ct_con_props (CS_SET、CS_DS_PROVIDER)、Server-Library では srv_props (CS_SET、SRV_DS_PROVIDER) を呼び出している場合)、アプリケーションは *libtcl.cfg* ファイルの [DIRECTORY] セクションを調べて一致するドライバを探し、該当のドライバをロードします。

ディレクトリ・ドライバと *libtcl*.cfg* の詳細については、「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照してください。

注意 手順 1 の内容は DB-Library には当てはまりません。DB-Library のディレクトリ・サービスを指定するには、手順 2 と 3 を使用してください。

2 クライアント・アプリケーションがディレクトリ・ドライバを指定していない場合、Client-Library、Server-Library、DB-Library は *libtcl.cfg* ファイルの [DIRECTORY] セクション内の最初のエントリにリストされているディレクトリ・ドライバをロードします。

3 次のいずれかに該当する場合、Client-Library、Server-Library、DB-Library は、*sql.ini* ファイルに戻り、このファイルを使用してサーバのアドレスを取得します。

- *libtcl.cfg* が存在しない。
- *libtcl.cfg* の [DIRECTORY] セクションにエントリがない。
- 指定されたディレクトリ・ドライバのロードに失敗した。
- CS_IFILE プロパティが ct_config によって設定されている場合、*libtcl*.cfg* はコンテキスト・レベルで上書きされる。

libtcl.cfg* ファイルを使用して、LDAP サーバ名、ポート番号、DIT ベース、ユーザ名、LDAP サーバへの接続を認証するためのパスワードを指定します。

libtcl.cfg* ファイルについて知っていなければならないことは、次のとおりです。

- *libtcl*.cfg* ファイルに指定されている値は、CS_* プロパティのデフォルトになります。これは、ct_con_props ルーチンによって設定されます。その特定の接続に ct_con_props ルーチンを明示的に設定することで、これらの値を上書きできます。
- *libtcl*.cfg* ファイルにパスワードとユーザ名のどちらも指定しない場合、接続は匿名になります。
- パスワードが “0x” で始まっている場合、接続プロパティではパスワードが暗号化されているものと見なします。詳細については、「パスワードの暗号化」(66 ページ) を参照してください。

- 64 ビットのプラットフォームでは、Open Client/Open Server には 32 ビットと 64 ビットの両方のバイナリがあります。32 ビット・アプリケーションと 64 ビット・アプリケーションの互換性を保つためには、*libtcl.cfg* と *libtcl64.cfg* ファイルの両方を編集しなければなりません。

libtcl.cfg* ファイルは、*%SYBASE%¥%SYBASE_OCS%¥ini* にあります。

アプリケーションが LDAP ディレクトリ・サービスを使用する仕組み

Sybase LDAP の機能を使用するには、ベンダが提供するマニュアルに従って、LDAP サーバをインストールし設定します。Sybase では、LDAP サーバを提供していません。Sybase では Netscape LDAP SDK クライアント・ライブラリを提供しており、Sybase Open Client/Open Server には、LDAP ドライバが含まれています。このドライバは、*%SYBASE%¥%SYBASE_OCS%¥dll* にあります。

Netscape LDAP SDK ライブラリは、*%SYBASE%¥%SYBASE_OCS%¥lib3p* にあります。Windows の PATH 環境変数で、このディレクトリを指定する必要があります。

LDAP ドライバが LDAP サーバに接続すると、サーバは匿名アクセスおよびユーザ名とパスワード認証の 2 つの認証方法に基づいて接続を確立します。

- 匿名アクセス － 認証情報を必要としないため、属性を設定する必要がありません。匿名アクセスは、一般には読み取り専用権限に使用します。
- ユーザ名とパスワード － LDAP URL の拡張機能として、*libtcl.cfg* ファイル (64 ビット・プラットフォームでは *libtcl64.cfg* ファイル) で指定するか (「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照)、Client-Library に対するプロパティ呼び出しで設定できます。ctlib を使用して LDAP サーバに渡されるユーザ名とパスワードは、Adaptive Server のログインに使用するユーザ名とパスワードとは別のものです。Sybase では、ユーザ名とパスワード認証を使用されることを強くおすすめします。

認証

クライアント・アプリケーションは、ホスト名とポート番号または IP アドレスを使用して、LDAP サーバへの接続を作成します。この接続は「バインド」と呼ばれ、安全ではない場合がありますが、ユーザ名とパスワード認証を使用することもできます。可能なアクセスのタイプは、サーバが決定します。

匿名接続

認証を必要としない接続は、匿名接続と呼ばれます。LDAP と Netscape Directory Services はデフォルトで匿名接続が可能です。

匿名アクセスの特徴は次のとおりです。

- 接続を確立するために、パスワードなどの認証情報を必要としません。
- 接続には、追加属性を設定する必要はありません。
- 一般的に、読み込み専用アクセスです。

ユーザ名とパスワード認証

書き込みを許可するアクセス・パーミッションに対しては、基本的なセキュリティの使用をおすすめします。ユーザ名とパスワードは、LDAP サーバへの接続に対して、基本レベルのセキュリティを提供します。ユーザ名とパスワードは、32 ビット・プラットフォームでは *libtcl.cfg* ファイルに、64 ビット・プラットフォームでは *libtcl64.cfg* ファイルに格納できます。また、Client-Library のプロパティで設定することもできます。

libtcl.cfg* ファイル、および設定ファイルでのパスワードの暗号化については、「付録 B 設定ファイル」を参照してください。

LDAP ディレクトリ・サービスの有効化

注意 LDAP だけが、リエントラント・ライブラリでサポートされています。LDAP ディレクトリ・サービスを使用してサーバに接続する場合は、isql ではなく、isql_r を使用してください。

❖ ディレクトリ・サービスを使用するように設定する

1 ベンダが提供するマニュアルに従って、LDAP サーバを設定します。

2 使用しているプラットフォームのパスに、LDAP ライブラリ・ディレクトリを追加します。次に例を示します。

```
PATH=%PATH%:%SYBASE%¥%SYBASE_OCS%¥lib3p
```

3 ディレクトリ・サービスを使用するように *libtcl*.cfg* ファイルを設定します。標準的な ASCII テキスト・エディタを使用して、次のように変更します。

- *libtcl*.cfg* ファイルの [DIRECTORY] エントリにある LDAP URL 行の行頭から、コメント・マーカのセミコロン (;) を削除します。
- [DIRECTORY] エントリに LDAP URL を追加します。サポートされている LDAP URL 値については、表 5-2 を参照してください。

注意 LDAP URL は、1 行で記述してください。

```
ldap=libsybdldap.dll ldap://host:port/ditbase??scope????
     bindname=username password
```

次に例を示します。

```
[DIRECTORY]
ldap=libsybdldap.dll ldap://huey:11389/dc=sybase,dc=com??
     one????bindname=cn=Manager,dc=sybase,dc=com secret
```

“*one*” は検索範囲を示し、DIT ベースの 1 つ下のレベルのエントリが取得されます。表 5-3 に、*ldapurl* 変数のキーワードの定義を示します。

表 *5-3: ldapurl* 変数

キーワード	説明	デフォルト	CS_* プロパティ
host (必須)	LDAP サーバを実行しているマシンのホスト名または IP アドレス	なし	
port	LDAP サーバが受信に使用しているポート番号	389	
ditbase (必須)	デフォルトの DIT ベース	なし	CS_DS_DITBASE
username	認証するユーザの DN (識別名)	NULL (匿名認証)	CS_DS_PRINCIPAL
password	認証されるユーザのパスワード	NULL (匿名認証)	CS_DS_PASSWORD

4　環境変数が PATH*%SYBASE%¥%SYBASE_OCS%¥lib3p* を指していることを確認します。

5　dsedit を使用して、LDAP サーバにサーバ・エントリを追加します。詳細については、「サーバ・エントリの作成と変更」(50 ページ) および 「ディレクトリ・サービスへのサーバの追加」(49 ページ) を参照してください。

LDAP を使った複数ディレクトリ・サービス

高可用性フェールオーバ保護には、複数のディレクトリ・サービスを指定できます。リストにあるディレクトリ・サービスがすべて LDAP サーバである必要はありません。次に例を示します。

```
[NT_DIRECTORY]
   ldap=libsybdldap.dll ldap://test:389/dc=sybase,dc=com
   dce=libddce.dll ditbase=/.:/subsys/sybase/dataservers
   ldap=libsybdldap.dll ldap://huey:11389/dc=sybase,dc=com
```

この例では、`test:389` への接続に失敗すると、接続は指定された DIT ベースの DCE ドライバにフェールオーバします。この接続にも失敗すると、`huey:11389` 上の LDAP サーバへの接続が試みられます。DIT ベースのフォーマットは、ベンダによってそれぞれ異なります。詳細については、『Open Client Client-Library/C リファレンス・マニュアル』を参照してください。

Microsoft Active Directory スキーマのインポート

ADAM (Active Directory Application Mode) インストール環境で提供されている ldifde.exe コマンドを使用して、*sybase.ldf* を AD (Active Directory) または ADAM インスタンスにインポートできます。ディレクトリ・スキーマをインポートするには、次の構文を使用して ADAM インストール環境から ldifde.exe コマンドを実行します。

```
ldifde -i -u -f sybase.ldf -s server:port -b username
domain password -j . -c "cn=Configuration,dc=X"
#configurationNamingContext
```

Sybase サーバ・エントリ用のコンテナの作成

Active Directory にスキーマを正常にインポートしたら、Sybase サーバ・エントリ用のコンテナを作成し、コンテナと子オブジェクトに適切な読み込みと書き込みのパーミッションを設定します。

たとえば、“CN=SybaseServers” という相対識別名 (RDN: Relative Distinguished Name) を持つコンテナをドメイン “mycompany.com” の Active Directory のルートに作成して、Sybase サーバ・エントリの保管と検索を行います。このコンテナのルート識別名 (rootDN) は、次のように *libtcl.cfg* ファイルに反映されます。

```
ldap=libsybdldap.dll ldap://localhost:389/
cn=SybaseServers,dc=mycompany,dc=com??...
```

Sybase サーバ・エントリの追加と修正を行うために、Active Directory にアカウント名 “Manager”、パスワード “secret” で専用のユーザ・アカウントを作成する場合、*libtcl.cfg* ファイルの完全なエントリは、次のようになります。

```
ldap=libsybdldap.dll
ldap://localhost:389/cn=SybaseServers,dc=mycompany,
dc=com????bindname=cn=Manager,cn=Users,dc=mycompay,
dc=com?secret
```

適切な読み込みと書き込みのパーミッションを設定したら、Sybase ユーティリティ・プログラム (dscp や dsedit など) を使用して、Active Directory 内の Sybase サーバ・エントリの保管、表示、修正を行うことができるようになります。

注意 Active Directory スキーマの拡張方法の詳細については、Microsoft Web サイトで「スキーマを拡張する」を検索してください。

SSL/TLS を使用した LDAP への接続

サポートされているすべてのプラットフォームで、SSL または TLS を使用して LDAP ディレクトリ・サーバへのセキュア接続を設定できます。クライアントと LDAP ディレクトリ・サーバとの間でセキュア接続を確立するには、次のいずれかの方法を使用します。

- *libtcl.cfg* ファイルに次の構文を入力して、LDAP サーバのセキュア・ポート (通常はポート番号 636) へのセキュア接続を確立します。

  ```
  [NT_DIRECTORY]
  ldap=libsybdldap.dll
  ldaps:// huey:636/dc=sybase,dc=com????
  bindname=cn=Manager,dc=Sybase,dc=com?secret
  ```

  ldaps:// を使用してポート番号を指定しない場合、ポート番号 636 がデフォルトで使用されます。

- StartTLS を使用して、標準の接続 (通常は、LDAP サーバのポート番号 389) をセキュア接続にアップグレードします。接続をアップグレードするには、*libtcl.cfg* ファイルに次の構文を入力します。

  ```
  [NT_DIRECTORY]
  ldap=libsybdldap.dll starttls
  ldap:// huey:389/dc=sybase,dc=com????
  bindname=cn=Manager,dc=Sybase,dc=com?secret
  ```

  ldap:// を使用してポート番号を指定しない場合、ポート番号 389 がデフォルトで使用されます。

詳細については、『Open Client Client-Library/C リファレンス・マニュアル』を参照してください。

DCE ディレクトリ・サービスの設定作業

Client-Library アプリケーションと Server-Library アプリケーションがディレクトリ・サービスを使用できるようにするには、次の作業を行います。

❖ ディレクトリ・サービスを使用するように設定する

1　ディレクトリ・サービスを設定するには、dsedit を使用してディレクトリ・サービスにターゲット・サーバのエントリを作成します。

dsedit の使用方法については、「第 8 章 dsedit の使用」を参照してください。

2　ocscfg を使用してディレクトリ・ドライバを設定します。

ディレクトリ・ドライバの設定方法については、「第 7 章 ocscfg の使用」を参照してください。

ディレクトリ・ドライバと *libtcl.cfg* の詳細については、「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照してください。

第 6 章

セキュリティ・サービスの使い方

Client-Library と Server-Library アプリケーションは、サード・パーティのセキュリティ・ソフトウェアが提供するセキュリティ・サービスを使用して、ユーザを認証し、ネットワーク上のマシン間で送信されるデータを保護することができます。

この章では、ネットワークベースのセキュリティが機能する仕組みと、この機能を使用するために必要な設定について説明します。

ネットワークベースのセキュリティの概要

分散クライアント／サーバ・コンピューティング環境では、不法侵入者が機密データを見たり改ざんしたりするおそれがあります。この可能性を低減するために、ネットワークベースのセキュリティでは、サード・パーティの分散セキュリティ・ソフトウェアを利用して、ユーザを認証し、ネットワーク上のマシン間で送信されるデータを保護します。

セキュリティ・メカニズム

Sybase では、「セキュリティ・メカニズム」を、接続時にセキュリティ・サービスを提供する外部ソフトウェアと定義しています。プラットフォームごとにそれぞれ異なるセキュリティ・メカニズムを使用できます。

Microsoft Windows NT LAN Manager (SSPI) と Kerberos は、Microsoft Windows 上のサーバとクライアントのためのセキュリティ・サービスを提供します。

sql.ini またはディレクトリ・サービスに、サーバがサポートするセキュリティ・メカニズムを指定できます。

- *sql.ini* エントリのオプションの secmech 行では、サーバがサポートするセキュリティ・メカニズムを指定します。
- ディレクトリ・サービス・エントリのオプションの secmech 属性では、サーバがサポートするセキュリティ・メカニズムを記述します。

クライアントはサーバのアドレスを取得するときに、クライアントが使用しているセキュリティ・メカニズムをサーバがサポートしているかどうかを確認できます。

- secmech 行または属性が指定されていて、セキュリティ・メカニズムがリストされている場合、使用できるのはそれらのセキュリティ・メカニズムだけです。
- secmech 行または属性がない場合は、すべてのセキュリティ・メカニズムを使用できます。
- secmech 行または属性が指定されていても、セキュリティ・メカニズムがリストされていない場合、サーバはどのセキュリティ・メカニズムもサポートしません。

セキュリティ・ドライバ

Sybase では、Client-Library と Server-Library がセキュリティ・メカニズムと通信できるようにするセキュリティ・ドライバを提供しています。Sybase の各セキュリティ・ドライバは、汎用インタフェースをセキュリティ・プロバイダのインタフェースにマップします。

接続でセキュリティ・メカニズムを使用するには、次の 2 つの条件のどちらも満たされている必要があります。

- クライアントとサーバは、互換性のあるセキュリティ・ドライバを使用する必要がある。たとえば、Microsoft Windows LAN Manager ドライバを使用するクライアントには、Microsoft Windows LAN Manager ドライバを使用するサーバが必要となる。
- クライアント・アプリケーションは、サーバに接続する前に、接続プロパティを設定してサービスを要求する必要がある。

セキュリティ・サービス

各セキュリティ・メカニズムは、クライアントとサーバ間にセキュア接続を確立するセキュリティ・サービスを提供します。各セキュリティ・サービスは特定のセキュリティ問題に対応しています。セキュリティ・サービスは大きく 2 つに分けられます。

- 認証サービス
- パケットごとのセキュリティ・サービス

セキュリティ・サービスの詳細については、『Open Client Client-Library/C リファレンス・マニュアル』を参照してください。

Client-Library アプリケーションは、セキュリティ・メカニズムのサービスを要求するように接続プロパティを設定します。Open Server アプリケーションは、クライアント・スレッドのプロパティを読み込んで、実行されているサービスを判別します。

LAN Manager セキュリティ・サービス

Windows LAN Manager サービスは、次のサービスを提供します。

- LAN Manager ユーザ・ネームスペースに基づくネットワーク認証
- データの整合性
- リプレイの検出
- 順序不整合の検出

Kerberos セキュリティ機能の詳細については、次の項を参照してください。

Kerberos セキュリティ・サービス

Kerberos セキュリティ・メカニズムは、次のサービスを提供します。

- ネットワーク認証
- 相互認証
- データの整合性
- データ機密保持
- リプレイの検出
- 順序不整合の検出
- クレデンシャル委任

これらのセキュリティ・サービスの詳細については、『Open Client Client-Library/C リファレンス・マニュアル』を参照してください。クライアント・アプリケーションがセキュリティ・サービスを使用する仕組みの概要については、「Client-Library とセキュリティ・サービス」(38 ページ) を参照してください。

CyberSafe Kerberos の設定

CyberSafe Kerberos セキュリティ・サービスを使用するクライアント・アプリケーションでは、次の点に注意してください。

- Open Client/Open Server 12.5 以降のシステムに、CyberSafe Kerberos ソフトウェアをインストールします。

- Client-Library アプリケーションの実行中は、*gssapi32.dll* ファイルがライブラリ・パスに含まれている必要があります。この DLL は Sybase が提供するものではなく、一部の CyberSafe Kerberos 製品に付属しています。この DLL が CyberSafe Kerberos 製品に付属していない場合は、CyberSafe Kerberos に連絡して *GSS-API* ライブラリを入手してください。
- *libtcl.cfg* 設定ファイルのセキュリテイ・セクションを設定します。
- ct_con_props を使用して、必要なセキュリティ機能を設定します。デフォルト・クレデンシャルを使用する場合は、クレデンシャル・プロパティを設定しないでください。
- アプリケーションに、サーバに接続するための既存のユーザ・クレデンシャルがあることを確認します。つまり、アプリケーションのユーザはクライアント・アプリケーションを実行する前に、CyberSafe Kerberos にログインする必要があります。これを行うには、シングル・サインオン機能または CyberSafe の kinit ユーティリティを使用します。
- ユーザ名を入力する場合、ユーザ名はそのユーザの既存のクレデンシャルと一致する必要があります。ユーザ名を入力しない場合、Client-Library はそのユーザの CyberSafe Kerberos クレデンシャルに対応するユーザ名を使用してサーバに接続します。
- 次に示す環境変数では、クレデンシャル・キャッシュ・ファイル、設定ファイル、レルム・ファイルへのパスを設定します。対応するファイルがデフォルト以外のディレクトリにある場合は、環境変数をファイルのフル・パスに設定します。
  - CSFC5CCNAME − クレデンシャル・キャッシュ・ファイル
  - CSFC5CONFIG − 設定ファイル
  - CSFC5REALMS − レルム・ファイル

  詳細については、CyberSafe Kerberos のマニュアルを参照してください。
- CyberSafe Kerberos セキュリティ・サービスを使用する Client-Library アプリケーションをコンパイルするときに、余分なフラグは必要ありません。
- Open Client/Open Server と CyberSafe Kerberos を設定したら、以下のコマンド (-U 引数と -P 引数なし) を使用して設定を検査できます。

```
isql -V
```

注意 ここで説明する作業の中には、CyberSafe Kerberos 管理ツールが必要なものもあります。詳細については、CyberSafe Kerberos のマニュアルを参照してください。

MIT Kerberos の設定

- MIT ソフトウェアのバージョン 2.6.5 以降をシステムにインストールし、設定します。
- *libtcl.cfg* 設定ファイルのセキュリテイ・セクションを設定します。
- ct_con_props を使用して必要なセキュリティ機能を設定するか、クレデンシャル・プロパティを設定しないでデフォルト・クレデンシャルを使用します。
- アプリケーションに、サーバに接続するための既存のユーザ・クレデンシャルがあることを確認します。つまり、アプリケーションのユーザは、kinit ユーティリティを使用して Kerberos 環境にログインしてから、クライアント・アプリケーションを実行します。
- ユーザ名を入力する場合、ユーザ名はそのユーザの既存のクレデンシャルと一致する必要があります。ユーザ名を入力しない場合、Client-Library はそのユーザのクレデンシャルに対応するユーザ名を使用してサーバに接続します。
- 環境変数 KRB5CCNAME は、クレデンシャル・キャッシュ・ファイルへのパスを設定します。対応するファイルがデフォルト以外のディレクトリにある場合は、環境変数をファイルのフル・パスに設定します。

  詳細については、マニュアルを参照してください。
- libgss キーワードを使用して、*libtcl.cfg* ファイルに MIT GSS ライブラリ (*gssapi32.dll*) を指定します。Kerberos ドライバに関して、フル・パスを指定することをおすすめします。
- Kerberos セキュリティ・サービスを使用する Client-Library アプリケーションをコンパイルするときに、余分なフラグは必要ありません。
- Open Client/Open Server と Kerberos を設定したら、isql を使用して設定を検査できます。

MIT Kerberos のクレデンシャル委任

Kerberos セキュリティ・ドライバは、MIT Kerberos GSS (Generic Security Services) ライブラリの使用時に、クレデンシャル委任をサポートしています。これにより、リモート・サーバとの接続を確立するときに、委任されたクライアント・クレデンシャルを使用する Open Server ゲートウェイ・アプリケーションを設定できます。

❖ クレデンシャル委任を使用してリモート・サーバとの接続を確立するには

これは、クレデンシャル委任の使用時に使用できる呼び出しシーケンスの一例です。*$SYBASE/OCS-15_0/sample/srvlibrary/connect.c* にある ctos の例には、ここで説明するプロパティの例が含まれています。

1 クライアント・アプリケーションは、次の構文を使用して、クレデンシャル委任を要求し、クレデンシャルをゲートウェイ接続に転送します。

```
ct_con_props(..., CS_SET, SRV_SEC_DELEGATION, ...)
```

2 ゲートウェイ・アプリケーションの接続ハンドラは、クライアントがクレデンシャル委任を要求しているかどうかをチェックします。

```
if (srv_thread_props(..., CS_GET,
     SRV_T_SEC_DELEGATION, ...))
    {...}
```

3 接続ハンドラは、委任されたクライアント・クレデンシャルを取得します。

```
srv_thread_props(..., CS_GET,
     SRV_T_SEC_DELEGCRED, ...)
```

4 クライアント・アプリケーションは、Client-Library 接続構造体内に、リモート・サーバへの接続に使用するための委任クレデンシャルを設定します。

```
ct_con_props(..., CS_SET, CS_SEC_CREDENTIALS, ...)
```

5 クライアント・アプリケーションは、ct_connect を使用してリモート・サーバへの接続を試みます。

isql と bcp オプション -Vd を使用して、クレデンシャル委任を要求することもできます。詳細については、『Open Client/Server プログラマーズ・ガイド補足 Microsoft Windows 版』を参照してください。

クレデンシャル委任の使用方法の詳細については、『Open Server Server-Library/C リファレンス・マニュアル』および『Open Client Client-Library/C リファレンス・マニュアル』を参照してください。

Windows セキュリティ SSPI の使用

Microsoft Windows セキュリティ・サポート・プロバイダ・インタフェース (SSPI: Security Support Provider Interface) によって Kerberos がサポートされている場合は、*libtcl.cfg* ファイルの *csfkrb5* エントリを編集して、*libsspiwrapper.dll* を GSS ライブラリとして指定します。

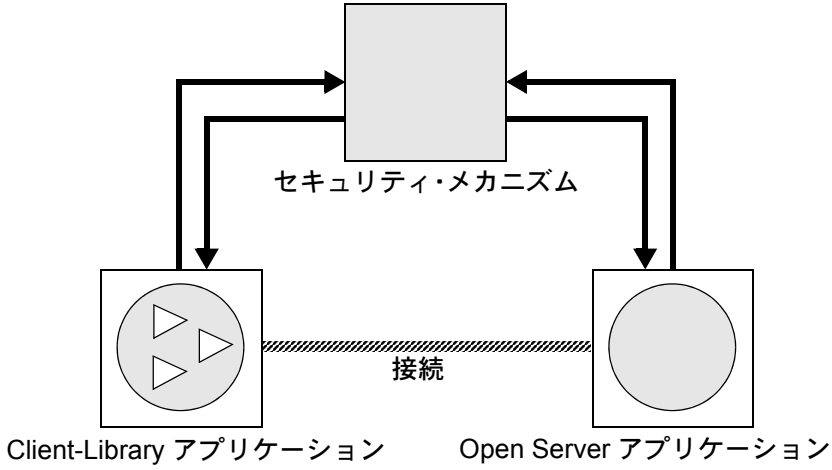
次に例を示します。

```
csfkrb5=LIBSKRB secbase=@REALM libgss=C:¥sybase¥OCS-15_0¥lib3p¥libsspiwrapper.dll
```

アプリケーションがセキュリティ・サービスを使用する仕組み

Client-Library アプリケーションと Server-Library アプリケーションは、セキュリティ・メカニズムを使用して、認証サービスとパケット単位セキュリティ・サービスを実行できます。セキュリティ・メカニズムは、Client-Library と Server-Library が情報を検証し合う情報交換所のようなものです。図 6-1 は、認証サービスとパケット単位セキュリティ・サービスの両方に当てはまります。

図 *6-1:* セキュリティ・メカニズムを使用する *Open Client* アプリケーションと *Open Server* アプリケーション



Open Client アプリケーションが認証サービスを要求した場合は、次の処理が行われます。

1　Client-Library はセキュリティ・メカニズムを使用してログインを検証します。セキュリティ・メカニズムはログイン・レコードまたはトークンを返します。セキュリティ・メカニズムは要求されたセキュリティ・サービスに基づいてログイン・トークンを作成します。

2　Client-Library は Open Server アプリケーションとのトランスポート接続を確立し、そのログイン・トークンを送信します。

3　Server-Library　は、セキュリティ・メカニズムを使用してクライアントのログイン・トークンを認証します。ログインが有効な場合、Open Server アプリケーションはセキュア接続を確立します。

Open Client アプリケーションがパケット単位セキュリティ・サービスを要求した場合は、次の処理が行われます。

1 Client-Library は、セキュリティ・メカニズムを使用して、Open Server アプリケーションに送信するデータ・パケットを用意します。セキュリティ・メカニズムは、要求されたセキュリティ・サービスに応じて、データを暗号化するか、データに対応する暗号化シグニチャを作成します。

2 Client-Library は Open Server アプリケーションにデータ・パケットを送信します。

3 Open Server はデータ・パケットを受信すると、セキュリティ・メカニズムを使用して必要な復号化と検証を実行します。

Client-Library のセキュリティ機能の詳細については、『Open Client Client-Library/C リファレンス・マニュアル』の「セキュリティ機能」を参照してください。

Client-Library とセキュリティ・サービス

セキュリティ・メカニズムのサービスを要求するように、Open Client アプリケーションの接続プロパティを設定できます。Client-Library は、接続で使用するセキュリティ・メカニズムとサービスを次のようにして決定します。

1 クライアント・アプリケーションにセキュリティ・ドライバの名前が指定されている場合は、Client-Library は *libtcl.cfg* ファイルを調べて、一致するドライバを探し、そのドライバをロードします。

2 クライアント・アプリケーションにセキュリティ・ドライバの名前が指定されていない場合は、Client-Library は *libtcl.cfg* にリストされている最初のセキュリティ・ドライバをロードします。

3 *libtcl.cfg* にセキュリティ・ドライバがリストされていない場合、サーバは正しいパスワードが入力されたかどうかでユーザを認証します。

Server-Library とセキュリティ・サービス

Open Server アプリケーションはクライアント接続要求のプロパティを読み込んで、使用するセキュリティ・メカニズムと実行するサービスを決定できます。

デフォルトでは、Open Server アプリケーションは、*libtcl.cfg* にリストされているセキュリティ・メカニズムをサポートします。システム管理者は、サーバのディレクトリ・エントリに secmech 属性を追加するか、Open Server アプリケーションの *sql.ini* ファイル・エントリに secmech 行を追加することによって、サポートするセキュリティ・メカニズムのリストをさらに制限できます。

Open Client アプリケーションが Open Server アプリケーションからのセキュリティ・セッションを要求すると、次の処理が行われます。

1　Server-Library は、クライアント接続要求と一緒に送信されたセキュリティ・トークンを読み込みます。セキュリティ・トークンには、クライアントが使用するセキュリティ・メカニズムのオブジェクト識別子が入っています。

2　Open Server アプリケーションの *sql.ini* エントリまたはディレクトリ・サービス・エントリに secmech 行／属性がリストされている場合は、Server-Library はこの secmech 行／属性を調べて、セキュリティ・トークンに指定されているオブジェクト識別子に対応する値を探します。対応する値が見つからない場合、接続要求は拒否されます。

3　Server-Library は *objectid.dat* を調べて、セキュリティ・メカニズムのローカル名に対応するオブジェクト識別子を探します。

objectid.dat の詳細については、「objectid.dat ファイル」(82 ページ) を参照してください。

4　Server-Library　は、セキュリティ・メカニズムのローカル名に対応するセキュリティ・ドライバをロードします。セキュリティ・ドライバは、*libtcl.cfg* にリストされています。

設定作業

Open Client/Open Server アプリケーションがセキュリティ・サービスを使用できるようにするには、ocscfg ユーティリティを使用してセキュリティ・ドライバを設定します。セキュリティ・ドライバの設定方法については、「第 7 章 ocscfg の使用」を参照してください。セキュリティ・ドライバと *libtcl.cfg* ファイルの詳細については、「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照してください。

オプションで、サーバがサポートしているセキュリティ・メカニズムを制限するには、次のいずれかを行います。

- アプリケーションが *sql.ini* ファイルを使用する場合は、dsedit ユーティリティを使用して、サーバの *sql.ini* ファイル・エントリに secmech 行を追加する。
- アプリケーションがディレクトリ・サービスを使用する場合は、dsedit ユーティリティを使用して、サーバのディレクトリ・サービス・エントリに secmech 属性を追加する。

ディレクトリ・サービスまたは *sql.ini* ファイルに情報を追加する方法については、「第 8 章 dsedit の使用」を参照してください。

第 7 章

ocscfg の使用

この章では、ocscfg ユーティリティを使用してローカル・マシンを設定する方法を説明します。

注意 ディレクトリ・サービスを設定している間に、dsedit にアクセスすることもできます。dsedit の使用方法については、「第 8 章 dsedit の使用」を参照してください。

ocscfg について

ocscfg を使用して、次の 3 種類の情報を設定できます。

- 環境変数
- ディレクトリ・ドライバ
- セキュリティ・ドライバ

ocscfg の起動

ocscfg は、[プログラム マネージャ]、DOS プロンプト、または [ファイル マネージャ] から起動できます。[スタート] メニューまたは Windows エクスプローラからも、ocscfg を起動できます。

- [プログラム マネージャ] から ocscfg を起動するには、Sybase プログラム・グループにあるアイコンをダブルクリックします。
- プロンプトから ocscfg を起動するには、次のように入力します。

```
ocscfg
```

- [ファイル マネージャ] から ocscfg を起動するには、次の手順に従ってください。

  a *%SYBASE%¥%SYBASE_OCS%¥bin* に移動します。*%SYBASE%* はインストール・ディレクトリです。

  b *ocscfg.exe* ファイルをダブルクリックします。

- [スタート] メニューから ocscfg を起動するには、[スタート] - [プログラム] - [Sybase] - [ocscfg] を選択します。

画面上部のタブの中から、実行する設定を選択します。次の表に、各タブで設定する機能を説明します。

タブ	機能
Environment	Sybase 関連の環境変数を設定する。ocscfg の起動時に、このダイアログ・ボックスがデフォルトで表示される。
Directory Service	*libtcl.cfg* ファイルにリストされているディレクトリ・ドライバを設定する。dsedit に接続する。
Security Service	*libtcl.cfg* ファイルにリストされているセキュリティ・ドライバを設定する。

環境変数の設定

[Environment] タブをクリックして、環境変数を設定するダイアログ・ボックスをアクティブにします。

SYBASE 環境変数の設定

SYBASE 環境変数を設定するには、次のいずれかを行います。

- [SYBASE] フィールドに Sybase インストール・ディレクトリのロケーションを入力する。
- [Browse] をクリックして、ローカル・ディレクトリ構造またはリモート・ディレクトリ構造を表示する。目的のディレクトリをダブルクリックして選択する。

注意 ocscfg は、SYBASE 環境変数を使用して *libtcl.cfg* ファイルを検索します。SYBASE 環境変数が正しく設定されていない場合、ocscfg は *libtcl.cfg* ファイルを見つけることができません。

その他の環境変数の設定

❖ **SYBASE 以外の環境変数を設定する**

1. [Environment Variables] ボックスから、設定する環境変数を選択します。選択した名前が [Variable Name] ボックスに表示されます。
2. [Value] ボックスに、選択した環境変数の値を入力します。
3. [Set] をクリックします。

詳細については、「付録 A 環境変数」を参照してください。

環境変数のクリア

❖ **SYBASE 以外の環境変数をクリアする**

1. [Environment Variables] ボックスから、設定する環境変数名を選択します。選択した名前が [Variable Name] ボックスに表示されます。
2. [Clear] をクリックします。

ディレクトリ・ドライバの設定

[Directory Services] タブをクリックして、ディレクトリ・ドライバを設定するダイアログ・ボックスを表示します。ocscfg ユーティリティは、ドライバ設定ファイル *libtcl.cfg* のロケーションをダイアログ・ボックスの上部に表示します。

ディレクトリ・ドライバ・エントリの追加

❖ **ディレクトリ・ドライバ・エントリを追加するには**

1. [Platform] ボックスから、使用しているプラットフォームを選択します。
2. ダイアログ・ボックスの下部にある [Add] をクリックします。[Add Directory Service Entry] ダイアログ・ボックスが表示されます。
3. [Directory Service Name] ボックスに、ディレクトリ・サービス名を入力します。ディレクトリ・サービスには、次の条件を満たす名前を付けることができます。
   - アルファベット、数字、アンダースコアだけで構成する。
   - 最大 64 文字

4 [Directory Service Driver] ボックスから、ドライバを選択します。

5 [Directory Service DIT base] ボックスから、DIT ベース値を選択します。DIT ベースは、ディレクトリ・サービスがサーバ・エントリの検索を始めるロケーションです。必要な構文については、「DIT ベース構文」(44 ページ)を参照してください。

6 [OK] をクリックします。

DIT ベース構文

ディレクトリ・ドライバ・エントリを追加または変更する場合は、DIT ベースを指定できます。DIT ベース構文は、選択したディレクトリ・ドライバによって異なります。

表 *7-1:* ディレクトリ・サービス *DIT* ベース構文

ディレクトリ・サービス	DIT ベース構文
Windows レジストリ	次に、レジストリの DIT ベース設定の例を 2 つ示す。 SOFTWARE¥SYBASE¥SERVER *machine_name*:SOFTWARE¥SYBASE¥SERVER 2 番目の例の *machine_name* は、ワークステーションのネットワーク名。 すべての DIT ベース・エントリは、*¥HKEY_LOCAL_MACHINE¥* を起点にする必要がある。DIT ベース・キー、および *¥HKEY_LOCAL_MACHINE¥* と DIT ベース・キーの間のすべてのキーについて、キー・エントリが必要。*¥HKEY_LOCAL_MACHINE¥¥SOFTWARE¥SYBASE* キーは、Sybase インストール・プログラムが作成する。上記の例の場合は、ユーザが *SERVER* キーを追加する必要がある。 Microsoft regedt32 ツールを使用して、必要なキーを作成する。レジストリ・パス名では、大文字と小文字を区別しない。

DIT ベースを指定しない場合、ディレクトリ・ドライバはデフォルト値 SOFTWARE¥SYBASE¥SERVER を使用します。

既存のディレクトリ・ドライバ・エントリの修正

❖ **既存のディレクトリ・ドライバ・エントリを修正するには**

1 [Platform] ボックスから、使用しているプラットフォームを選択します。

2 [Directory Service Name] フィールドで、修正するディレクトリ・サービス名を選択します。

3 ダイアログ・ボックスの下部にある [Edit] をクリックします。[Edit Directory Service Entry] ダイアログ・ボックスが表示されます。

4　必要に応じて、ディレクトリ・サービス名、ドライバ、DIT ベースを更新します。

5　[OK] をクリックします。

ディレクトリ・ドライバ・エントリの削除

❖ **ディレクトリ・ドライバ・エントリを削除するには**

1　[Platform] ボックスから、使用しているプラットフォームを選択します。

2　[Directory Service Name] フィールドで、削除するディレクトリ・サービス名を選択します。

3　[Delete] をクリックします。

ディレクトリ・ドライバのアクティブ化

ocscfg は、[Active Directory Service] ボックスにアクティブ・ディレクトリ・ドライバを表示します。最初にリストされているドライバがアクティブ・ドライバです。

❖ **ディレクトリ・ドライバのアクティブ化**

1　[Directory Service Name] フィールドで、アクティブにするディレクトリ・サービス名を選択します。

2　[Set Active] をクリックします。

セキュリティ・ドライバの設定

[Security Service] タブをクリックして、セキュリティ・ドライバを設定するダイアログ・ボックスを表示します。ocscfg ユーティリティは、ドライバ設定ファイル *libtcl.cfg* のロケーションをダイアログ・ボックスの上部に表示します。

セキュリティ・ドライバ・エントリの追加

❖ **セキュリティ・ドライバ・エントリを追加するには**

1　[Platform] ボックスから、使用しているプラットフォームを選択します。

2　[追加] をクリックします。[Add Security Service Entry] ダイアログ・ボックスが表示されます。

3 [Local Name] ボックスにセキュリティ・サービス名を入力します。

セキュリティ・サービスのローカル名は、*objectid.dat* にあるエントリに対応していなければなりません。詳細については、「objectid.dat ファイル」(82 ページ) を参照してください。

4 [Security Service Driver] ボックスから、ドライバを選択します。

5 [OK] をクリックします。

既存のセキュリティ・ドライバ・エントリの修正

❖ **既存のセキュリティ・ドライバ・エントリを修正するには**

1 [Platform] ボックスから、使用しているプラットフォームを選択します。

2 [Local Name] フィールドで、設定するセキュリティ・サービス名を選択します。

3 ダイアログ・ボックスの下部にある [Edit] ボタンをクリックします。[Edit Security Service Entry] ダイアログ・ボックスが表示されます。

4 必要に応じて、セキュリティ・サービス名とドライバを更新します。

5 [OK] をクリックします。

セキュリティ・ドライバ・エントリの削除

❖ **セキュリティ・ドライバ・エントリの削除**

1 [Platform] ボックスから、使用しているプラットフォームを選択します。

2 [Local Name] フィールドで、設定するセキュリティ・サービス名を選択します。

3 [Delete] をクリックします。

デフォルト・セキュリティ・ドライバの設定

ocscfg ユーティリティは、[Default Local Name] フィールドにデフォルト・セキュリティ・ドライバを表示します。最初にリストされているドライバがデフォルト・ドライバです。

❖ **デフォルト・セキュリティ・ドライバの設定**

1 [Local Name] フィールドで、設定するセキュリティ・サービス名を選択します。

2 [Set Default] をクリックします。

第 8 章

dsedit の使用

この章では、dsedit を使用してディレクトリ・サービスまたは *sql.ini* を設定する方法について説明します。

dsedit の使用

dsedit ユーティリティを使用すると、ディレクトリ・サービスや *sql.ini* を設定できます。

dsedit は、プログラム・アイコン、DOS プロンプト、または [ファイル マネージャ] から起動できます。dsedit は、[スタート] メニューや [エクスプローラ] からも起動できます。

- プログラム・アイコンから dsedit を起動するには、Sybase プログラム・グループの dsedit アイコンをダブルクリックします。
- DOS プロンプトから dsedit を起動するには、次のように入力します。

  ```
  dsedit
  ```

  次のコマンド・ライン引数を指定できます。

引数	説明
-d dsname	ディレクトリ・サービスの接続先を指定する。*dsname* には、*libtcl.cfg* ファイルにリストされているディレクトリ・サービスのローカル名が入る。 -d dsname 引数を指定していない場合、dsedit は最初に表示されるダイアログ・ボックスにディレクトリ・サービス・オプションのリストを表示する。
-l path	*libtcl.cfg* ファイルが *%SYBASE%¥%SYBASE_OCS%¥ini* 以外の場所に位置する場合は、そのパスを指定する。 *%SYBASE%¥%SYBASE_OCS%¥ini* 以外にある *libtcl.cfg* ファイルを使用する場合にだけ、この引数を使用する。

- [ファイル マネージャ] または [エクスプローラ] から dsedit を起動するには、次の手順に従ってください。

  a *%SYBASE%¥%SYBASE_OCS%¥bin* ディレクトリに移動します。

  b *dsedit.exe* をダブルクリックします。

- [スタート] メニューから dsedit を起動するには、[スタート] - [プログラム] - [Sybase] - [dsedit] を選択します。

セッションのオープン

[Select Directory Service] ダイアログ・ボックスを使用して、ディレクトリ・サービスのセッションをオープンできます。次のいずれかを使用してセッションをオープンできます。

- *libtcl.cfg* ファイルにドライバがリストされている任意のディレクトリ・サービス
- *sql.ini*

セッションをオープンするには、次のいずれかを行います。

- [DS Name] ボックスで、接続するディレクトリ・サービスのローカル名をダブルクリックします。
- 接続するディレクトリ・サービスのローカル名をクリックし、[OK] をクリックします。

注意 dsedit は、SYBASE 環境変数を使用して *libtcl.cfg* を探します。SYBASE 環境変数が正しく設定されていない場合、dsedit は *libtcl.cfg* を見つけることができません。

セッション番号とディレクトリ・サービスのローカル名が、ヘッダ・バーに表示されます。

追加セッションのオープン

dsedit ユーティリティを使用すると、複数のセッションをオープンできます。

❖ **追加セッションのオープン**

1 [File] メニューから [Open Directory Service] を選択します。

[Select Directory Service] ボックスが表示されます。

2 接続するディレクトリ・サービスのローカル名をダブルクリックするか、ディレクトリ・サービス名をクリックし、[OK] をクリックします。

複数のセッションをオープンすることによって、ディレクトリ・サービス間でエントリをコピーできます。詳細については、「サーバ・エントリのコピー」(53 ページ) を参照してください。

セッションのアクティブ化

セッションをアクティブにしてから、作業を始めてください。セッションをアクティブにするには、次のいずれかを行います。

- セッション・ウィンドウをクリックします。
- Windows メニューからセッションを選択します。

上部の dsedit ヘッダ・バーに、アクティブになっているセッションが表示されます。

ディレクトリ・サービスへのサーバの追加

警告！ ほとんどの LDAP サーバには、ディレクトリ・エントリを追加するための ldapadd ユーティリティがありますが、代わりに、一般のツールからは提供されない組み込みのセマンティック・チェックがある dsedit を使用することをおすすめします。

dsedit は、*libtcl*.cfg* ファイルと *sql.ini* ファイル内のサーバ・エントリを追加、削除、修正できるグラフィカル・ユーティリティです。LDAP URL を *libtcl*.cfg* ファイルに追加してから、LDAP サーバ・エントリの追加、削除、または修正を行ってください。詳細については、「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照してください。

❖ **dsedit を使用してディレクトリ・サービスにサーバを追加するには**

次のように、dsedit を使用してディレクトリ・サービスにサーバを追加します。

1. Microsoft Windows の [スタート] メニューから、[プログラム] - [Sybase] - [dsedit] の順に選択します。
2. サーバの一覧から [LDAP] を選択して、[OK] をクリックします。
3. [Add New Server Entry] をクリックします。

4 次のように入力します。

- サーバ名 – 必須です。
- セキュリティ・メカニズム – オプションです。セキュリティ・メカニズムの OID の一覧は、*%SYBASE%¥ini¥objectid.dat* にあります。
- HA サーバ名 – オプションです。高可用性フェールオーバ・サーバがある場合は、その名前です。

5 [Add New Network Transport] をクリックします。

- ドロップダウン・リストからトランスポート・タイプを選択します。
- ホスト名を入力します。
- ポート番号を入力します。

6 [OK] を 2 回クリックして、dsedit ユーティリティを終了します。

サーバ・エントリを表示するには、Web ブラウザで次の URL を入力します。

```
ldap://host:port/ditbase??one
```

次に例を示します。

```
ldap://huey:11389/dc=sybase,dc=com??one
```

注意 Microsoft Internet Explorer では、LDAP URL は認識されません。

サーバ・エントリの作成と変更

ディレクトリ・サービスまたは *sql.ini* のセッションをオープンすると、そのセッションに対応するサーバ・エントリの追加、変更、名前の変更、削除が可能になります。

セッションに対応するサーバ・エントリが、[Server] ボックスに表示されます。サーバ・エントリをクリックし、選択してください。

各サーバ・エントリは、一連の属性で構成されます。表 8-1 に示すサーバ・エントリの属性と属性値が、ダイアログ・ボックスの右側に表示されます。

表 *8-1:* サーバの属性

属性名	値のタイプ	説明	デフォルト値
Server Entry Version	整数	サーバ・オブジェクト定義のバージョン・レベル。 この属性は、オブジェクト定義の今後の変更を識別するために用意されている。	15501
Server Name	文字列	サーバの名前。	該当なし
Server Service	文字列	サーバが提供するサービスの説明。 意味のある説明ならどのような内容でも有効。	ASE
Server Status	整数	サーバの実行状態。 有効な値は次のとおり。 1 － アクティブ 2 － 停止 3 － 失敗 4 － 不明	4
Server Address	文字列	サーバの 1 つまたは複数のアドレス。 アドレスのフォーマットはプロトコルによって異なる。また、複数のフォーマットを使用できるプロトコルもある。オプションは次のとおり。 • TCP/IP (以下の 2 つのフォーマットがある) 1. *computer_name,port_number* 2. *ip-address,port_number* • 名前付きパイプ *pipe_name*：すべてのパイプ名に、プレフィックスとして “¥pipe” が必要。サーバ・パイプはローカルのみ。 (ローカル) *¥pipe¥sql¥query* (リモート) *¥¥computer_name¥pipe¥sql¥query* • IPX/SPX (以下の 3 つのフォーマットがある) 1. *server_name* 2. *net_number,node_number,socket_number* 3. *server_name, socket_number*	該当なし
Server HAfailover (オプション)	文字列	高可用性フェールオーバ・サーバが構成されている場合はその名前。	該当なし
Server Security (オプション)	文字列	サーバがサポートするセキュリティ・メカニズムを指定するオブジェクト識別子 (OID) の文字列。この属性はオプション。省略した場合、Open Server は Open Server が対応するセキュリティ・ドライバを持つ任意のセキュリティ・メカニズムにクライアントが接続できるようにする (詳細については、「Server-Library とセキュリティ・サービス」を参照)。 OID の詳細については、「objectid.dat」を参照。	該当なし

サーバ・エントリの追加

❖ **サーバ・エントリの追加**

1 [Server Object] メニューから、[Add] を選択します。[Input Server Name] ボックスが表示されます。

2 [Server Name] ボックスに、サーバ名を入力します。

3 [OK] をクリックします。

サーバ・エントリが [Server] ボックスに表示されます。サーバのアドレスを指定するには、エントリを修正します。

サーバ・エントリの修正

❖ **サーバ・エントリの修正**

1 [Server] ボックスで、サーバ・エントリをクリックします。

2 [Attributes] ボックスで、変更する属性をクリックします。

3 [Server Object] - [Modify Attribute] を選択します。ダイアログ・ボックスが表示され、属性の現在の値が表示されます。

4 属性の新しい値を入力するか、ドロップ・ダウン・リストから値を選択します。各属性の詳細については、表 8-1 (51 ページ) を参照してください。

5 [OK] をクリックします。

サーバ・エントリの名前の変更

❖ **サーバ・エントリの名前の変更**

1 [Server] ボックスで、サーバ・エントリをクリックします。

2 [Server Object] - [Rename] を選択します。[Input Server Name] ボックスが表示されます。

3 [Input Server Name] ボックスに、サーバ・エントリの新しい名前を入力します。

4 [OK] をクリックします。

エントリの削除

❖ **サーバ・エントリを削除する**

1 [Server] ボックスで、サーバ・エントリをクリックします。

2 [Server Object] - [Delete] を選択します。

ping コマンドの使用

❖ **ping を使用してネットワーク接続を確認するには**

1 [Server] ボックスで、サーバ・エントリをクリックします。

2 [Server Object] - [Ping] コマンドを選択します。[Ping] ダイアログ・ボックスが表示されます。

3 ping を送るアドレスをクリックします。

4 [Ping] をクリックします。

接続に成功したか、失敗したかを知らせるメッセージ・ボックスが表示されます。接続に失敗した場合は、「接続障害のトラブルシューティング」(55 ページ) を参照してください。

サーバ・エントリのコピー

dsedit ユーティリティでは、1 つのセッション内、または複数のセッション間でサーバ・エントリをコピーできます。*sql.ini* ファイルからディレクトリ・サービスにエントリをコピーすることも可能です。

セッション内のエントリのコピー

❖ **現在のセッション内でエントリをコピーする**

1 [Server] ボックスで 1 つ以上のサーバ・エントリをクリックします。複数のエントリを選択する場合は、[Shift] キーを使用します。

2 [Copy] ボタン (メニュー・バーの下) をクリックするか、[Edit] - [Copy] を選択します。

3 [Paste] ボタン (メニュー・バーの下) をクリックするか、[Edit] - [Paste] を選択します。

dsedit は、コピーしたサーバ・エントリにバージョン番号を表す *_n* を付加します。[Server Object] - [Rename] コマンドを使用して、コピーしたサーバ・エントリの名前を変更できます。詳細については、「サーバ・エントリの名前の変更」(52 ページ) を参照してください。

セッション間のエントリのコピー

❖ **セッション間でサーバ・エントリをコピーする**

1 ディレクトリ・サービスまたは *sql.ini* を使用して、エントリのコピー先のセッションをオープンします。

セッションをオープンするには、[File] - [Open Directory Service] を選択します。詳細については、「追加セッションのオープン」(48 ページ) を参照してください。

2 コピーするエントリのあるセッションの [Server] ボックスで、1 つ以上のサーバ・エントリをクリックします。複数のエントリを選択する場合は、[Shift] キーを使用します。

3 サーバ・エントリをコピーするには、[Copy] をクリックするか、[Edit] - [Copy] を選択します。サーバ・エントリを切り取るには、[Cut] をクリックするか、[Edit] - [Cut] を選択します。

4 サーバ・エントリを貼り付けるセッションをアクティブにします。セッションをアクティブにする方法については、「セッションのアクティブ化」(49 ページ) を参照してください。

5 [Paste] をクリックするか、[Edit] - [Paste] を選択します。

[Server Object] - [Rename] コマンドを使用して、コピーしたサーバ・エントリの名前を変更できます。詳細については、「サーバ・エントリの名前の変更」(52 ページ) を参照してください。

dsedit の終了

dsedit を終了するには、[File] - [Exit] を選択します。

第 9 章

dsedit のトラブルシューティング

この章では、Sybase のディレクトリ・サービス・ユーティリティである dsedit を使用して、Client-Library アプリケーションと、Adaptive Server または Open Server アプリケーションとの間のネットワーク接続を検査する方法について説明します。

dsedit の実行方法

dsedit を使用することによって、Net-Library ソフトウェアが正しくインストールされ、Adaptive Server または Open Server アプリケーションに接続できることを検証できます。dsedit は、Client-Library の ct_connect ルーチンや Net-Library の対話と同じように機能しますが、Adaptive Server または Open Server アプリケーションで有効なユーザ名を持つ必要はありません。

Net-Library のインストールが完了すると、いつでも dsedit を実行できます。

サーバの接続をテストするには、dsedit を使用しているサーバに対して ping を実行します。詳細については、「第 8 章 dsedit の使用」を参照してください。

接続障害のトラブルシューティング

アプリケーションがサーバに接続できなかった場合は、dsedit を実行します。dsedit が表示したメッセージを調べれば、問題解決に役立つ情報が得られます。

問題によっては、dsedit では診断できないものもあります。これらは、多くの場合、Net-Library とネットワーク・ソフトウェア間の接続ではなく、Adaptive Server または Open Server の設定に関わる問題です。これらの問題については、「dsedit は成功したが他のアプリケーションが失敗した場合」(57 ページ) を参照してください。

dsedit が失敗した場合

dsedit が接続に失敗した場合は、Net-Library の次の基本的な稼働条件がすべて満たされているかどうかを確認します。

- Adaptive Server または Open Server がサーバ・マシンで稼働している。
- ユーザ PC とサーバ・マシンの間にネットワーク・ハードウェア接続が存在する。
- ユーザの PC がハードウェアとソフトウェアの最小要件を満たしている。
- ネットワーク・ベンダのソフトウェアがユーザの PC にインストールされ、稼働している。
- *sql.ini* ファイルの接続情報が正しい。

上記の稼働条件が満たされている場合は、表示されたメッセージを調べて、dsedit がどの時点で失敗したかを判断します。

dsedit がサーバに接続できない場合は、メッセージ・ボックスが表示されます。

dsedit は接続情報を見つけたが、サーバから応答がないことが通知された場合、次の点を確認してください。

- サーバが稼働していることを確認します。サーバを実行しているマシンにアクセスできる場合は、isql を使用して、サーバにログインしてみてください。マシンにアクセスできない場合には、必要なサーバが稼働していることをシステム管理者に確認します。
- ネットワークのソフトウェアとハードウェアが正しく設定されていることを確かめます。たとえば、コネクタ、プラグなどをチェックし、ネットワーク・ソフトウェアが実行されていることを確認してください。
- メッセージ・ボックスにネットワーク・エラー・メッセージが表示されていないかどうかを確かめたり、システム・イベント・ログを見てエラーの有無を調べます。
- システム管理者に連絡して、サーバを実行しているマシンに接続できるように接続情報の値が正しくなっているかどうかを確認します。または、ネットワーク・ソフトウェアに含まれているユーティリティを使用して、自分でこれを確認してください。

自分で問題を分析できない場合には、Sybase とのコンタクト・パーソンを通して、Sybase 製品の保守契約を結んでいるサポート・センタに連絡し、問題を報告してください。詳細については、「Sybase 製品の保守契約を結んでいるサポート・センタへの問い合わせに必要な情報」(57 ページ) を参照してください。

dsedit は成功したが他のアプリケーションが失敗した場合

dsedit はエラーを報告していないが、他のアプリケーションが実行できない場合は、次の手順に従います。

- アプリケーションがデフォルト・サーバを使用するかどうかを確認します。ct_connect ルーチンのサーバ名を渡す場合は、dsedit リストから該当するサーバを選択していることを確認してから、接続をテストします。
- Adaptive Server または Open Server アプリケーションに有効なユーザ・ログイン名を持っていること、およびデータベースとテーブルに関するパーミッションがアプリケーションを実行するのに必要なパーミッションと一致していることを確認します。
- isql ユーティリティを使用して、Adaptive Server または Open Server アプリケーションにアクセスできることを確認します。isql の詳細については、『Open Client/Server プログラマーズ・ガイド補足』を参照してください。
- Adaptive Server または Open Server アプリケーションを実行しているマシンで、isql を使用して、アプリケーションが使用するデータベースとテーブルが実在するかどうかを確認します。Adaptive Server または Open Server アプリケーションを実行するマシンに対するアクセス権がない場合や、isql についてよくわからない場合には、データベース管理者に連絡して確認します。

Sybase 製品の保守契約を結んでいるサポート・センタへの問い合わせに必要な情報

Net-Library 製品に問題が発生して、Sybase 製品の保守契約を結んでいるサポート・センタに連絡する必要がある場合は、次の情報を提供できるようにしてください。

- ネットワーク・ソフトウェアの名前とバージョン番号
- ネットワーク・ソフトウェアが稼働するオペレーティング・システムの名前とバージョン番号
- 接続するサーバが稼働するオペレーティング・システムの名前とバージョン番号
- 接続するサーバのバージョン番号
- Net-Library DLL の日付とサイズ。この情報を入手するには、DIR コマンドを実行して、DLL が含まれているファイル・リストを表示します。

一般的な質問

- 質問：新しいバージョンの Sybase DLL を入手したのに、ソフトウェアが引き続き旧バージョンの動作をする。

  答え：マシンにある DLL が 1 コピーだけであることを確認してください。同じ名前の DLL がもう 1 つある場合は、パスをチェックして、どのディレクトリが最初にリストされているかを調べてください。誤って古い方のバージョンの DLL をロードしている可能性があります。

- 質問：cs_ctx_alloc が失敗する。

  答え：*sybinit.err* をオープンして cs_ctx_alloc が失敗した理由の詳細説明を探します。*sybinit.err* は、アプリケーションがインストールされているディレクトリにあります。

- 質問：ct_init が失敗する。

  答え：*sybinit.err* ファイルをオープンして cs_init が失敗した理由の詳細説明を探します。このファイルはアプリケーションがインストールされているディレクトリにあります。

  libtcl.cfg にリストされているドライバがすべてインストールされていることと、それらのファイルへのパスが *wsybset.bat* にリストされていることを確認してください。

- 質問：isql または dsedit を実行すると、Sybase が提供するものではない DLL が見つからないことを示す「ファイル・エラー」メッセージが表示される。

  答え：この DLL はネットワーク・ベンダの DLL と思われます。このメッセージは、ネットワークが正しくインストールされていないことを示します。

付 録　A

環境変数

この付録では、設定情報となる環境変数を説明します。

接続に使用する環境変数

Open Client/Open Server は、接続プロセス中に表 A-1 に示す環境変数を使用します。

表 *A-1:* 接続に使用する環境変数

変数	値	デフォルト	使用箇所
DSLISTEN	*sql.ini* またはディレクトリ・サービスにリストされている Open Server アプリケーションの名前。 DSLISTEN が設定されていない場合、Open Server はデフォルト値 “SYBASE” を使用する。	SYBASE	Open Server DSLISTEN は Open Server アプリケーションが初期化時にサーバを指定していない場合にだけ使用する。
DSQUERY	*sql.ini* またはディレクトリ・サービスにリストされているターゲット・サーバの名前。 DSQUERY が設定されていない場合、Open Client はデフォルト値 “SYBASE” を使用する。	SYBASE	Open Client DSQUERY は Open Client アプリケーションがサーバの名前を指定していない場合にだけ使用する。
SYBASE	Sybase インストール・ディレクトリのロケーション。	“sybase” ユーザのホーム・ディレクトリ	Open Client と Open Server
SYBASE_OCS	Open Client/Open Server 製品のホーム・ディレクトリ。	OCS-15_0	Open Client と Open Server
PATH	Open Client/Open Server 製品が実行ファイルと DLL を探すときに検索するディレクトリ・パス。	実行プログラム	Open Client と Open Server

ローカライゼーションで使用する環境変数

Open Client/Open Server は、ローカライゼーション時に表 A-2 に示す環境変数を使用します。

表 *A-2:* ローカライゼーションで使用する環境変数

環境変数	設定するロケール名の内容	使用
LC_ALL	メッセージ、データ型変換、ソートに使用する言語、文字セット、照合順。	初期ローカライゼーション、カスタム・ローカライゼーション
LANG	メッセージ、データ型変換、ソートに使用する言語、文字セット、照合順。 Open Client/Server 製品は、LC_ALL 環境変数を見つけることができない場合には LANG 環境変数を検索する。	初期ローカライゼーション
LC_COLLATE	文字データのソートと比較を行うときに使用する照合順 (ソート順)。	カスタム・ローカライゼーション
LC_CTYPE	データ型変換に使用する文字セット。	カスタム・ローカライゼーション
LC_MESSAGE	メッセージに使用する言語。	カスタム・ローカライゼーション
LC_TIME	日付と時刻のフォーマット、ネイティブ言語での名前、月と日の省略形などの日時文字列に使用する日付と時刻のデータ表現。	カスタム・ローカライゼーション

LC_* 環境変数は POSIX 標準の環境変数であり、Sybase 以外のアプリケーションでも使用できます。*locales.dat* ファイルには、Sybase 以外のアプリケーションの環境変数で使用するのと同じロケール名がリストされていることを確認してください。

設定で使用する環境変数

Open Client/Open Server 製品は、設定プロセス中に表 A-3 に示す環境変数を使用します。

表 *A-3:* 設定で使用する環境変数

環境変数	説明	使用
SYBOCS_CFG	デフォルトの外部設定ファイル・パスの *%SYBASE%¥%SYBASE_OCS%¥ini¥ocs.cfg* を上書きする。	実行時
SYBOCS_DBVERSION	実行時に §DB-Library バージョン・レベルを外部から設定する。DB-Library は、DB-Library の初期化段階でこの変数を使用して環境変数を取得し、その環境変数値をバージョン・レベルとして保存する。 詳細については、『Open Client DB-Library/C リファレンス・マニュアル』を参照。	実行時
SYBOCS_DEBUG_FLAGS	特定の診断サブシステムを有効にする。複数のデバッグ・オプションを有効にするには、変数にカンマで区切ったフラグのリストを指定する。 デバッグの詳細については、『Open Client Client-Library C リファレンス・マニュアル』を参照。	実行時
SYBOCS_DEBUG_LOGFILE	診断を記録するログ・ファイルを指定する。この変数を設定しない場合、メッセージは stdout に書き込まれる。 『Open Client Client-Library/C リファレンス・マニュアル』を参照。	実行時

付 録　B

設定ファイル

この付録では、Open Client/Open Server 製品が設定情報を入手するときに使用するファイルについて説明します。

設定ファイルについて

設定ファイルはインストール時に Sybase インストール・ディレクトリ構造内のデフォルト・ロケーションに作成されます。

表 B-1 は、Open Client/Open Server 製品が使用する設定ファイルを示します。

表 *B-1:* 設定ファイルの名前とロケーション

ファイル名	説明	ロケーション	参照箇所
libtcl.cfg	このドライバ設定ファイルには、ディレクトリ、セキュリティ、ネットワークの各ドライバに関する情報と、必要な初期化情報が格納されている。	*%SYBASE%¥%SYBASE_OCS%¥ini* **注意** CS_LIBTCL_CFG プロパティを使用して、*libtcl.cfg* ファイルへの代替パスを指定します。	「libtcl.cfg ファイルと libtcl64.cfg ファイル」(64 ページ) を参照。 『Open Client/Server Common Libraries リファレンス・マニュアル』も参照。
sql.ini	このファイルには、ファイルにリストされている各サーバのネットワーク情報とセキュリティ情報が格納されている。このファイルは *libtcl.cfg* ファイルのバックアップとしても使用される。	*%SYBASE%¥ini*	「sql.ini ファイル」(69 ページ) を参照。
objectid.dat	文字セット、照合順、セキュリティ・メカニズムのロケール名にグローバル・オブジェクト識別子をマップする。	*%SYBASE%¥ini*	「付録 C ローカライゼーション」を参照。
ocs.cfg	ランタイム設定ファイルによって、実行時に一部の Open Client アプリケーションの値を変更できる。	*%SYBASE%¥%SYBASE_OCS%¥ini*	「ocs.cfg ファイル」(73 ページ) を参照。

libtcl.cfg ファイルと *libtcl64.cfg* ファイル

libtcl.cfg ファイルと *libtcl64.cfg* ファイル (総称して *libtcl*.cfg* ファイル) は、Open Client/Open Server 製品で使用する次の 2 つのタイプのドライバの情報を含むドライバ設定ファイルです。

- ディレクトリ・ドライバ
- セキュリティ・ドライバ

ドライバは、Open Client/Open Server ソフトウェアに外部サービス・プロバイダとの汎用インタフェースを提供する Sybase ライブラリです。Open Client と Open Server は、ドライバを使用することによって、複数のサービス・プロバイダを容易にサポートできます。

ネットワーク、ディレクトリ、またはセキュリティ・ドライバをロードすると、Open Client と Open Server は、*libtcl*.cfg* を読み込みます。*libtcl.cfg* のエントリは、Open Client/Open Server 製品にドライバの名前とそのドライバの初期化情報を提供します。

libtcl.cfg* ファイルの目的は、設定情報 (Open Client/Open Server と Open Client/Open Server ベースのアプリケーション用のドライバ、ディレクトリ、セキュリティ・サービスなど) を提供することです。*libtcl.cfg* と *libtcl64.cfg* は、いずれも 64 ビット・プラットフォーム上で提供されます。dsedit や srvbuild などの (64 ビット・プラットフォーム上の) 32 ビット・アプリケーションは *libtcl.cfg* ファイルで設定情報を探し、64 ビット・アプリケーションは *libtcl64.cfg* ファイルで設定情報を探します。

libtcl.cfg* ファイルは、*sql.ini* ファイルと LDAP ディレクトリ・サービスのどちらを使用するかを指定します。*libtcl*.cfg* ファイルに LDAP が指定されている場合は、サーバへの接続時に **-l** パラメータを渡すことによってアプリケーションが *libtcl*.cfg* ファイルを明示的に上書きしないかぎり、*sql.ini* ファイルは無視されます。

libtcl.cfg ファイルは、*%SYBASE%¥%SYBASE_OCS%¥ini* ディレクトリにあります。

libtcl.cfg のレイアウト

libtcl.cfg は、ドライバのタイプごとに複数のセクションに分けられています。ocscfg は、次のようなセクション見出しを作成します。

セクション見出し	説明
NT_DIRECTORY	Microsoft Windows ディレクトリ・ドライバのリスト
SECURITY	Microsoft Windows セキュリティ・ドライバのリスト

注意 これらのセクションを特定の順序に並べる必要はありません。

ディレクトリ・ドライバ

ディレクトリ・ドライバ・エントリの構文は次のとおりです。

```
provider=driver ditbase
```

各要素の意味は次のとおりです。

- *provider* は、ディレクトリ・サービスのローカル名です。この要素には、アルファベット、数字、アンダースコアだけで構成される、64 文字以内の任意の名前を付けることができます。
- *driver* は、LIBSYBDREG という Microsoft Windows レジストリ・ドライバの名前です。ドライバのデフォルト・ロケーションは *%SYBASE%¥%SYBASE_OCS%¥dll* です。
- *ditbase* はディレクトリ・サービスがサーバ・エントリの検索を始めるロケーションです。*ditbase* の構文は、ディレクトリ・サービス・プロバイダによって異なります。

ディレクトリ・サービス	DIT ベース構文

Microsoft Windows レジストリ	レジストリの DIT ベース設定の例：
	`ditbase=SOFTWARE¥SYBASE¥SERVER`
	`ditbase=machine_name:SOFTWARE¥SYBASE¥SERVER`
	2 番目の例の *machine_name* は、ワークステーションのネットワーク名。
	すべての DIT ベース・エントリは、*¥HKEY_LOCAL_MACHINE¥* を起点にする必要がある。DIT ベース・キー、および *¥HKEY_LOCAL_MACHINE¥* と DIT ベース・キーの間のすべてのキーについて、キー・エントリが必要。
	その他の必要なキーを作成するには、Microsoft regedt32 ツールを使用する。レジストリ・エントリは大文字と小文字を区別しない。

DCE ディレクトリ・サービス ditbase 構文

DCE ディレクトリ・サービスを使用する場合は、*libtcl.cfg* ファイルの DIT ベース情報に次の構文を使用します。

```
ditbase=/.:/dce_cell_name
```

次に例を示します。

```
ditbase=/.:/subsys/sybase/dataservers
```

DIRECTORY セクションの LDAP エントリ

最も簡単なフォームでは、LDAP ディレクトリ・サービスは、次のようなフォーマットで指定されます。

```
[DIRECTORY]
ldap=libsybdldap.dll ldapurl
```

ここでは、*ldapurl* は次のように定義されています。

```
ldap://host:port/ditbase
```

次の LDAP エントリは上記と同じ属性を使用していますが、匿名接続であり、LDAP サーバが読み込み専用アクセスを許可している場合にだけ動作します。

```
ldap=libsybdldap.dll
ldap://test:389/dc=sybase,dc=com
```

LDAP URL への拡張機能として、*libtcl*.cfg* ファイルでユーザ名とパスワードを指定すると、接続時にパスワード認証が有効になります。

ユーザ名を設定するには、次のように入力します。

```
if (ct_con_props(conn, CS_SET, CS_DS_PRINCIPAL, ldapprincipal,
     strlen(ldapprincipal), (CS_INT *)NULL) != CS_SUCCEED)
{
     ...
}
```

パスワードを設定するには、次のように入力します。

```
if (ct_con_props(conn, CS_SET, CS_DS_PASSWORD, ldappassword,
     strlen(ldappassword), (CS_INT *)NULL) != CS_SUCCEED)
{
     ...
}
```

パスワードの暗号化

libtcl.cfg ファイルと *libtcl64.cfg* ファイルのエントリは、人間が判読できるフォーマットです。Sybase では基本的なパスワードの暗号化のために、pwdcrypt ユーティリティを提供しています。pwdcrypt は、キーボード入力時に使用すると、暗号化された値を生成する単純なアルゴリズムです。生成された値をパスワードの代用として使用できます。pwdcrypt は *%SYBASE%¥%SYBASE_OCS%¥bin* にあります。

❖ パスワードの暗号化

1 Open Client/Open Server (OCS) ディレクトリから、コマンド・プロンプトに次のように入力します。

```
bin/pwdcrypt
```

2 パスワードの入力を要求されたら、パスワードを 2 回入力します。次のように、pwdcrypt が暗号化されたパスワードを生成します。

```
0x01312a775ab9d5c71f99f05f7712d2cded2i8d0ae1ce78868d0e8669313d1bc4c706
```

3　標準的な ASCII テキスト・エディタを使用して、暗号化されたパスワードをコピーして *libtcl*.cfg* ファイルに貼り付けます。暗号化の前に、ファイル・エントリが次のように表示されます。

注意　LDAP URL は、1 行で記述してください。

```
ldap=libsybdldap.dll
ldap://dolly/dc=sybase,dc=com????bindname=cn=Manager,dc=sybase,
dc=com?secret
```

4　パスワードを、暗号化した文字列に置き換えます。

```
ldap=libsybdldap.dll
ldap://dolly/dc=sybase,dc=com????bindname=cn=Manager,dc=sybase,dc=com?
0x01312a775ab9d5c71f99f05f7712d2cded2i8d0ae1ce78868d0e8669313d1bc4c706
```

警告！　パスワードが暗号化されている場合でも、ファイル・システム・セキュリティを使用してパスワードを保護してください。

セキュリティ・ドライバ

セキュリティ・ドライバ・エントリの構文は、次のとおりです。

```
provider=driver init_string
```

各要素の意味は次のとおりです。

- *provider* は、セキュリティ・メカニズムのローカル名です。セキュリティ・メカニズムのローカル名は、オブジェクト識別子ファイル *%SYBASE%¥ini¥objectid.dat* にリストされています。

  objectid.dat ファイルの詳細については、「objectid.dat ファイル」(82 ページ) を参照してください。

- *driver* はドライバの名前です。ドライバのデフォルト・ロケーションは *%SYBASE%¥%SYBASE_OCS%¥dll* です。指定できる *driver* の値は次のとおりです。

ドライバ名	説明
libsybsdce.dll	Gradient DCE ドライバ
libsybsmssp.dll	Microsoft Windows LAN Manager ドライバ
libsybskrb.dll	Kerberos セキュリティ ドライバ

- *init_string* はドライバの初期化文字列です。この要素はオプションです。*init_string* の値はドライバによって異なります。

Kerberos ドライバには、*init_string* に、セキュリティ・プリンシパル名のオプションの修飾子を指定します。*init_string* の構文は次のとおりです。*realm* は、レルム情報が使用できない場合にプリンシパル名に追加される値です。レルム名の先頭がアットマーク (@) でない場合は、プリンシパル名とレルム情報の間にスラッシュ (/) が挿入されます。

```
secbase=realm
```

セキュリティ・サービス初期化構文

Open Client と Open Server は、Kerberos セキュリティ・ドライバをサポートしています。Kerberos セキュリティ・ドライバを使用するには、次のいずれかを行います。

- ocscfg ユーティリティを使用して、Security Services に変更を加えます。
- *%SYBASE%¥%SYBASE_OCS%¥ini* ディレクトリの *libtcl.cfg* を直接編集します。

ocscfg ユーティリティの使用

ocscfg を使用するには、[Security Services] タブを開き [Add] をクリックします。ダイアログ・ボックスに次のように入力します。

- *Local Name*：*csfkrb5* を入力するか、*objectid.dat* ファイルで Kerberos ドライバに割り当てた名前を入力します。
- *Security Service Driver*：[Security Service Init String] メニューから LIBSYBSKRB を選択します。

これらの 2 つの項目を入力したら、[OK] をクリックします。エントリが、[Security Services] タブのダイアログ・ボックスに表示されます。

libtcl.cfg の編集

libtcl.cfg ファイルを直接編集する場合は、Kerberos セキュリティ・ドライバの *provider* 値を *csfkrb5* に設定するか、*objectid.dat* ファイルで Kerberos セキュリティ・ドライバに割り当てた値に設定します。*driver* の値を LIBSYBSKRB に設定します。次のフォームの *libtcl.cfg* に初期化文字列を設定する必要があります。

```
secbase=@your_realm_name
```

your_realm_name は、Kerberos プリンシパルを持つレルムです。Microsoft Windows ではこのエントリが必要です。次に例を示します。

```
[SECURITY]
csfkrb5=LIBSYBSKRB secbase=@SYBASE_CYBER_REALM
```

objectid.dat ローカライゼーション・ファイルの詳細については、「付録 C ローカライゼーション」を参照してください。

DCE セキュリティ・サービス初期化構文

DCE セキュリティ・サービスを使用する場合は、*libtcl.cfg* ファイルの初期化文字列情報で次の構文を使用します。

```
secbase=/.../dce_cell_name
```

次に例を示します。

```
secbase=/.../dsatestcell
```

libtcl.cfg の例

```
[NT_DIRECTORY]
 ntreg_dsa=LIBDREG  ditbase=software¥sybase¥serverdsa

[SECURITY]
 NTLM=LIBSMSSP
```

libtcl.cfg の編集

ocscfg を使用して、*libtcl.cfg* ファイルにドライバを設定します。ocscfg の使い方については、「第 7 章 ocscfg の使用」を参照してください。

sql.ini ファイル

sql.ini ファイルには、サーバのネットワーク・ロケーションに関する情報が含まれています。Open Client と Open Server は、機能を限定したディレクトリ・サービスとして *sql.ini* を使用します。*sql.ini* は、外部ディレクトリ・サービスに障害が発生した場合のデフォルトとしても機能します。デフォルトでは、Open Client/Open Server 製品は *%SYBASE%¥ini* ディレクトリで *sql.ini* ファイルを探します。

- Open Client は *sql.ini* ファイルのエントリの query 行に指定されているネットワーク情報を使用してサーバに接続する。
- Open Server は *sql.ini* ファイルのエントリの master 行に指定されているネットワーク情報を使用して、クライアント接続要求を受信する。

アプリケーションでは、Open Client/Open Server 製品が *sql.ini* を探す場所として、別のロケーションを指定できます。詳細については、『Open Client Client-Library/C リファレンス・マニュアル』の ct_config、および『Open Server Server-Library/C リファレンス・マニュアル』の srv_props を参照してください。dsedit を使用して、*sql.ini* を編集します。dsedit の使用方法については、「第 8 章 dsedit の使用」を参照してください。

sql.ini エントリ

sql.ini ファイルのエントリは、次のようなフォームになります。

```
[SERVERNAME]
service_type=driver,address
secmech=mechanism1,...,mechanismn
```

各要素の意味は次のとおりです。

- *SERVERNAME* は Open Client または Open Server が読み込む *sql.ini* エントリを認識するときに使用するエイリアスです。*SERVERNAME* は、文字 (ASCII の a-z、A-Z) で始まる必要があります。文字、数字、アンダースコアだけで構成される 11 文字以内の名前を指定できます。
- *service_type* には、接続のタイプを指定します。

  Microsoft Windows では、*service_type* のオプションは次のとおりです。

  - master 行には “master”。サーバ・アプリケーションがクライアント・クエリを受信するときに使用する。
  - query 行には “query”。クライアント・アプリケーションがサーバを探すときに使用する。

  sql.ini エントリの master 行と query 行には、同じ情報が含まれています。dsedit は、各エントリに両方のタイプの行を作成します。結果のエントリはクライアントとサーバの両方が使用できます。
- *driver* には、接続に使用するネットワーク・ドライバの名前が入ります。ネットワーク・ドライバのリストは次のとおりです。

ドライバ	説明
NLWNSCK	Winsock TCP/IP ドライバ
NLMSNMP	名前付きパイプ・ドライバ
NLNWLINK	SPX/IPX ドライバ

- *address* には、指定されたサーバのネットワーク・アドレスが入ります。アドレス情報のフォーマットは接続に使用するネットワーク・プロトコルによって異なります。*address* のオプションは次のとおりです。

プロトコル	フォーマット	例
TCP/IP	次の 2 種類がある。 1. *computer_name,port_number* 2. *ip-address port_number*	TEST,8877 130.214.30.25,8877
名前付きパイプ	*pipe_name*：すべてのパイプ名に、プレフィックスとして “¥pipe” が必要。サーバ・パイプはローカルのみ。 (ローカル) *¥pipe¥sql¥query* (リモート) *¥¥computer_name¥pipe¥sql¥query*	
IPX/SPX	次の 3 種類がある。 1. *server_name* 2. *net_number,node_number,socket_number* 3. *server_name, socket_number*	TEST 16,1,83BD TEST,83BD

- “secmech” は、サーバがサポートするセキュリティ・メカニズムをリストするときに使用する識別子です。“secmech” 行はオプションです。

  secmech 行の詳細については、「セキュリティ・メカニズム」(31 ページ) を参照してください。

- *mechanism1,..., mechanism* はサーバがサポートしているセキュリティ・メカニズムです。セパレータとしてカンマ (“,”) を使用することで、複数のセキュリティ・メカニズムを指定できます。

  セキュリティ・メカニズムはオブジェクト識別子としてリストされます。オブジェクト識別子は、グローバル・オブジェクト識別子ファイル内のセキュリティ・メカニズムのローカル名にマップした、グローバルにユニークな数字列です。

  オブジェクト識別子の詳細については、「objectid.dat ファイル」(82 ページ) を参照してください。

sql.ini の例

次の表に、各プロトコルの *sql.ini* の例を示します。

プロトコル	例

TCP/IP	`[SYBASE]`
	`master=NLWNSCK,TEST,8877`
	`query=NLWNSCK,TEST,8877`

名前付きパイプ	`[SYBASE]` `master=NLMSNMP,¥PIPE¥SQL¥`QUERY` `query=NLMSNMP,¥¥TEST¥PIPE¥SQL¥`UERY`

複数の接続サービス・エントリ

サーバは複数のネットワークでクライアントを受信できます。クライアントは実行時に複数のネットワークでサーバに接続できます。

複数のネットワークで受信するサーバ

サーバが複数のネットワークで受信するには、サーバの *sql.ini* ファイルを編集して、サーバが受信するネットワークごとに 1 つずつ “master” エントリを作成します。たとえば、サーバ “MYSERVER” に、次のような *sql.ini* エントリがあるとします。

```
MYSERVER
    master = NLWNSCK,mercury,1234
    master = NLNWLINK,my_mercury_spx
```

あるサーバが *sql.ini* を解析し、サーバ名 MYSERVER を見つけると、受信 TCP/IP 接続の場合は TCP/IP アドレス “mercury,1234” で受信し、受信 IPX/SPX 接続の場合は SPX バイナリ・アドレス “my_mercury_spx” で受信します。

複数のネットワークで接続するクライアント

クライアントが複数のネットワークで接続するには、クライアントの *sql.ini* ファイルを編集して、クライアントが接続するネットワークごとに 1 つずつ “query” エントリを作成します。たとえば、サーバ “SERVER99” の *sql.ini* エントリに、次のような “query” サービスがあるとします。

```
SERVER99
    query = NLWNSCK,mercury,1234
    query = NLWNSCK,plato,9876
    query = NLMSNMP,¥¥plato¥pipe¥sql¥query
    query = NLNWLINK,my_mercury_spx
```

Open Client アプリケーションは、まず “mercury,1234” でサーバへの接続を試みます。この試行に失敗すると、“plato,9876” でサーバへの接続を試みます。この試行も失敗すると、名前付きパイプ・プロトコルを使用して、“¥¥plato¥pipe¥sql¥query” でサーバへの接続を試みます。最後の試行として、IPX/SPX プロトコルを使用して、“my_mercury_spx” でサーバへの接続を試みます。この最後の試行にも失敗すると、Open Client はエラーを返します。

ocs.cfg ファイル

ランタイム設定ファイル *ocs.cfg* は、Client-Library アプリケーションが以下を設定するために使用します。

- プロパティ値
- サーバ・オプション値
- サーバ機能
- デバッグ・オプション

ocs.cfg を使用することによって、アプリケーションで値を設定するルーチンを呼び出す必要がなくなります。*ocs.cfg* を使用する利点は、コードを再コンパイルしなくても、アプリケーションの設定値を変更できることです。

デフォルトでは、Client-Library は *ocs.cfg* を読み込みません。Client-Library がこのファイルを使用できるように、アプリケーションでプロパティを設定する必要があります。

ファイル構文と、ファイルに設定できるプロパティについては、『Open Client Client-Library/C リファレンス・マニュアル』の「ランタイム設定ファイルの使い方」を参照してください。

構文の詳細については、『Open Client Client-Library/C リファレンス・マニュアル』を参照してください。

付 録　C

ローカライゼーション

ローカライゼーションとは、特定の言語を使用して、その言語を使用する国の慣習に従って実行できるように、アプリケーションを初期化するプロセスです。

この付録では、システム設定の観点からローカライゼーションとローカライゼーション・ファイルを説明します。ローカライゼーションに関するプログラミングの問題については、『Open Client/Open Server 開発者用国際化ガイド』を参照してください。

ローカライゼーション・プロセスの概要

Open Client/Open Server アプリケーションは、次の 2 通りの方法でローカライズできます。

- 初期ローカライゼーション値の使用
- 初期ローカライゼーション値とカスタム・ローカライゼーション値の使用

すべての Open Client/Open Server アプリケーションは初期ローカライゼーション値を使用します。これは、実行時に決定されます。

さらに、アプリケーション実行時の特定の時点でローカライズする必要がある場合、Open Client/Open Server アプリケーションでは、カスタム・ローカライゼーション値も使用できます。カスタム・ローカライゼーション値は、実行時に設定された初期ローカライゼーション値を上書きします。

ローカライゼーション時に使用する環境変数

Open Client と Open Server は環境変数を使用して、*locales.dat* ファイルでどのロケール名を探すかを決定します。初期ローカライゼーション値を設定する場合は、Open Client と Open Server は次の環境変数を検索します。

- LC_ALL
- LANG (LC_ALL が設定されていない場合)

カスタム・ローカライゼーション値を設定する場合は、Open Client と Open Server は次の環境変数も検索することがあります。

- LC_ALL
- LC_COLLATE
- LC_TYPE
- LC_MESSAGE
- LC_TIME

カスタム・ローカライゼーション時にアプリケーションが使用する環境変数については、『Open Client/Open Server 開発者用国際化ガイド』を参照してください。

上記にリストした環境変数の詳細については、「付録 A 環境変数」を参照してください。

ローカライズされたアプリケーションを実行する前に、次の点に注意してください。

- *locales.dat* ファイルに、アプリケーションが使用するローカライゼーション値を反映したエントリが入っていることを確認してください。入っていない場合は、該当するエントリを追加してください。
- アプリケーションが使用するローカライゼーション・ファイルがインストールされていることを確認してください。
  - ローカライズされたメッセージ・ファイルは、*%SYBASE%¥locales¥message* ディレクトリにあります。
  - 照合順ファイルは、*%SYBASE%¥charsets* ディレクトリにあります。

  Open Client/Open Server 製品には、1 つの言語、および 1 つ以上の文字セットとソート順をサポートするローカライゼーション・ファイルが付属しています。

ローカライゼーション・ファイル

Open Client/Open Server アプリケーションは、実行時に外部ファイルからローカライゼーション情報を取り出します。これらのファイルは、Sybase リリース・ディレクトリの以下の 3 つのディレクトリに格納されています。

- *locales* ディレクトリには、次のディレクトリとファイルが入っています。
  - ロケール名を言語、文字セット、照合順にマップする *locales.dat* ファイル。
  - Open Client/Server 用のローカライズされたエラー・メッセージが入っている *message* サブディレクトリ。
  - 以前のリリースの Open Client/Server ソフトウェアとの互換性のために用意されている *language_name* サブディレクトリ。このディレクトリには、ローカライズされたメッセージ・ファイルが文字セット別に編成されて入っています。
  - システム管理ユーティリティ用のエラー・メッセージ・ファイルが入っている、*unicode* ディレクトリ。
- *charsets* ディレクトリには、サポートされている各文字セットのサブディレクトリが入っています。各サブディレクトリには、文字セットのソート・ファイルと変換ファイルが入っています。
- *ini* ディレクトリには、次のファイルが入っています。
  - オブジェクトのグローバル識別子をローカル・プラットフォーム固有の名前にマップする *objectid.dat* ファイル。

すべての Open Client/Open Server 製品には、最低 1 つの言語と、1 つまたは複数の文字セットと照合順をサポートするファイルが含まれています。インストール時に、これらのファイルは Sybase ホーム・ディレクトリ構造の正しいロケーションにロードされます。

Open Client アプリケーションまたは Open Server アプリケーションを設定するときは、上記のディレクトリにユーザ・サイトとユーザ・アプリケーションの適切なファイルが入っていることを確認してください。

locales ディレクトリ

locales ディレクトリには、アプリケーションがローカライゼーション情報をロードするときに使用するファイルが入っています。また、言語固有のメッセージ・ファイルも入っています。

locales.dat ファイル

%SYBASE%¥locales ディレクトリにある *locales.dat* には、プラットフォーム固有のロケール情報が Sybase 独自のフォーマットで入っています。このファイルは、言語、文字セット、照合順とロケール名を対応させます。

警告！ isql は、サーバとの通信時にクライアント文字セットのデフォルトとして iso_1 を使用します。isql を使用する場合は、データの破壊を防ぐために、次のいずれかを実行して文字セットを修正してください。

- *locales.dat* ファイルのセクションに isql.german.cp850 などの新しいエントリを追加し、-J isql オプションを指定して isql を呼び出します。
- LANG=isql を設定し、クライアント文字セットを cp850 に変更します。
- 表示文字セットが iso_1 に変更されるように、mode con cp SELECT=1250 などのコマンドを発行してから isql を呼び出します。

---

locales.dat ファイルの使い方

Open Client/Open Server アプリケーションは、*locales.dat* を使用して、ロードするローカライゼーション情報を決定します。*locales.dat* は Open Client/Open Server アプリケーションのためのローカライゼーション情報を格納していますが、ローカライズされた実際のメッセージまたは文字セットの情報は含まれていません。

locales.dat のセクションとエントリ

locales.dat は、プラットフォーム固有のセクションで構成され、各セクションには事前定義されたロケール定義エントリが含まれています。これらのエントリはプラットフォームによって異なりますが、すべてのセクションに “default” ロケールを定義するエントリが含まれています。

ロケール定義エントリの形式は、次のとおりです。

```
locale = locale_name, language_name, charset_name
[,sortorder_name]
```

各要素の意味は次のとおりです。

- *locale_name* は、ロケール定義の名前です。*locale_name* のデフォルト値は、ベンダ指定であり、POSIX 用語規定に基づいています。*locales.dat* ファイルの最後にあるコメントには、ロケール名の POSIX 値がリストされます。
- *language_name* は、Sybase 製品が言語を認識するときに使用するサブディレクトリ名です。
- *charset_name* は、Sybase 製品が文字セットを認識するときに使用するサブディレクトリ名です。

- *sortorder_name* は、Sybase 製品が照合順を認識するときに使用するファイル名です (オプション)。

次の *locales.dat* ファイル・エントリでは、フランス語のロケールを指定しています。このロケールではソート順が指定されていないため、デフォルトのソート順である「バイナリ」が使用されます。

```
locale = fr.FR.88591, french, iso_1
```

locales.dat の例

locales.dat ファイルの次の部分は、*locales.dat* ファイルのプラットフォーム固有のセクションを示しています。

```
[NT]
locale = enu, us_english, cp1252
locale = fra, french, cp1252
locale = deu, german, cp1252
locale = default, us_english, cp1252
```

locales.dat の編集

locales.dat の事前定義されたエントリがユーザのニーズに合わない場合は、テキスト・エディタを使用してファイルを編集します。

警告！ 編集を行う前に、元の *locales.dat* のコピーを作成してください。コピーを作成しておくと、編集したファイルで問題が発生した場合に役立ちます。また、プラットフォームのエントリを調べて、エントリが既にあるかどうかも確認してください。

次の操作ができます。

- “default” ロケール定義を変更します。
- ロケール定義を追加します。
- Sybase 以外のソフトウェアが使用するロケール名に合わる。たとえば、次のように Sybase で事前定義されているロケール名は “fr” です。

  ```
  locale = fr, french, iso_1
  ```

  Sybase 以外のアプリケーションで、LC_ALL 環境変数の値として “french” が必要な場合は、ロケール名を次のように変更します。

  ```
  locale = french, french, iso_1
  ```

locales.dat ファイルに新しいエントリを追加したり、既存のエントリを変更するには、次の手順に従ってください。

1　*locale_name* の値を選択します。

2　*language_name* の値を決定します。

Sybase 言語モジュールをインストールすると、Sybase ディレクトリ・ツリーの *locales¥message* ディレクトリに言語のサブディレクトリが作成されます。*language_name* は、このサブディレクトリの名前と一致している必要があります。

3 *charset_name* の値を決定します。

Sybase の言語モジュールをインストールすると、Sybase ディレクトリ・ツリーの *charsets* ディレクトリに、サポートされている文字セットごとにサブディレクトリが作成されます。*charset_name* は、これらのサブディレクトリ名のいずれかと一致している必要があります。

4 バイナリ以外のソート順が必要な場合は、*sortorder_name* の値を決定します。

charsets¥charset_name サブディレクトリには、文字セットのソート順ファイル (**.srt*) が入っています。*sortorder_name* は、これらのファイル名 (*.srt* を除く) のいずれかと一致する必要があります。

5 *locales.dat* ファイルの該当するプラットフォーム固有セクションで、該当するエントリを入力または変更します。

変更後は、次の作業を行ってください。

- ローカライゼーション環境変数 (LC_ALL、LC_CTYPE、LC_MESSAGE、LC_TIME、LANG) を適切に更新します。
- 新しいロケール名を既に追加しており、既存のアプリケーションの cs_locale 呼び出しでこの新しい名前を使用するようにしたい場合は、アプリケーションを適切に編集し、再コンパイルします。

アプリケーションがエントリを使用しなくなっても、*locales.dat* からそのエントリを削除する必要はありません。エントリを削除する場合は、どのアプリケーションもそのエントリを使用していないことを確認してください。

ローカライズされたメッセージ・ファイル

警告！ ローカライズされたメッセージ・ファイルは編集しないでください。

ローカライズされたメッセージ・ファイルには、特定の言語で記述した製品メッセージが含まれています。これらのメッセージ・ファイル (*%SYBASE%¥locales¥message¥ language_name* ディレクトリの **.loc* ファイル) を使用することで、Open Client/Open Server アプリケーションはさまざまな言語でメッセージを生成できるようになります。

すべての Open Client/Server 製品には、英語 (us_english) のメッセージ・ファイルが入っています。他の言語をサポートするためのファイルが含まれている場合もあります。

新しい言語モジュールを購入してインストールした場合、インストール・プロセスで *language_name* サブディレクトリが追加され、新しい言語のメッセージ・ファイルが格納されます。

メッセージ・ファイル名はプラットフォームによって異なることもありますが、ほとんどの場合、次のような名前になります。

- *cslib.loc* － CS-Library メッセージ
- *ctlib.loc* － Client-Library メッセージ
- *oslib.loc* － Server-Library メッセージ
- *blklib.loc* – Bulk Library メッセージ
- *bcp.loc* – Bulk Copy メッセージ
- *esql.loc* – Embedded SQL メッセージ

Open Client/Open Server のすべてのメッセージ・ファイルは、Unicode ISO 10646 UTF-8 文字セットを使用します。

Open Client/Open Server 製品は、必要に応じてメッセージを UTF-8 から他の文字セットに変換します。

charsets ディレクトリ

charsets ディレクトリには、サポートされている各文字セットの照合順ファイルと、Unilib® が使用する変換ファイルが格納された *unicode* ディレクトリが入っています。

照合順ファイル

警告！ 照合順ファイルは編集しないでください。

システムが文字をソートする順序は、照合順またはソート順と呼ばれます。

Open Client/Open Server 製品には、さまざまな照合順をサポートするファイルが用意されています。*%SYBASE%¥charsets* ディレクトリにあるこれらのファイルは、プラットフォームによって異なる場合もありますが、通常は次のファイルが含まれています。

- *binary.srt*
- *dictionary.srt*
- *noaccents.srt*

- *nocase.srt*
- *nocasepref.srt*

照合順は、*locales.dat* ファイル・エントリに指定されています。*locales.dat* ファイル・エントリに照合順が指定されていない場合は、バイナリ・ソート順を使用します。

照合順の詳細については、『Open Client/Open Server 開発者用国際化ガイド』を参照してください。

Unicode 変換ファイル

Unicode 変換ファイルには、UTF-8 形式の Unicode (ISO 10646) 文字セットの変換設定情報が含まれています。これらの変換ファイルは、Sybase がサポートする各文字セットで利用できます。

ini ディレクトリ

ini ディレクトリには、グローバル・オブジェクト識別子ファイル (*objectid.dat*) が入っています。

objectid.dat ファイル

グローバル・オブジェクト識別子ファイル *objectid.dat* は、オブジェクトのローカル名をユニークなグローバル・オブジェクト識別子に対応させます。

オブジェクト識別子は、ドットで区切った一連の正の整数値です。オブジェクト識別子は、国際標準化団体である CCITT と ISO が定義したネーミング・ツリーに基づいています。

objectid.dat のロケーション

objectid.dat は、*Sybase_home¥locales* ディレクトリにあります。

objectid.dat のセクションとエントリ

objectid.dat は、オブジェクト・クラスごとに 1 つのセクションで構成されています。

オブジェクト・クラス・エントリのフォームは次のとおりです。

```
[Object Class]
    object_identifier local_name1, ..., local_namen
```

各パラメータの意味は、次のとおりです。

- *Object Class* はセクション識別子です。
- *object_identifier* はグローバルにユニークなオブジェクト識別子です。
- *local_name1,..., local_namen* は、カンマで区切ったオブジェクト識別子に対応するローカル名です。

objectid.dat の例

objectid.dat ファイルの次の部分は、*objectid.dat* のセクションを示します。

```
[charset]
     1.3.6.1.4.1.897.4.9.1.1 = iso_1
     1.3.6.1.4.1.897.4.9.1.2 = cp850
     1.3.6.1.4.1.897.4.9.1.3 = cp437
     1.3.6.1.4.1.897.4.9.1.4 = roman8
     1.3.6.1.4.1.897.4.9.1.5 = mac

[collate]
     1.3.6.1.4.1.897.4.9.3.50 = binary
     1.3.6.1.4.1.897.4.9.3.51 = dictionary
     1.3.6.1.4.1.897.4.9.3.52 = nocase
     1.3.6.1.4.1.897.4.9.3.53 = nocasepref
     1.3.6.1.4.1.897.4.9.3.54 = noaccents

[secmech]
     1.3.6.1.4.1.897.4.6.1 = dce, dcesecmech
     1.3.6.1.4.1.897.4.6.3 = NTLM, N, ntsecmech
     1.3.6.1.4.1.897.4.6.6 = csfkrb5, kerberos
```

注意　オブジェクトのローカル名を変更する場合は、テキスト・エディタを使用して *objectid.dat* を適宜編集します。

付 録　D

Open Client/Open Server の SSL (Secure Socket Layer)

この付録では、Open Client/Open Server の SSL サポートと、SSL プロトコルを使用するために必要なシステム設定作業について説明します。

Open Client と Open Server のセキュリティ・サービス・アーキテクチャの概要については、「第 6 章 セキュリティ・サービスの使い方」を参照してください。

SSL ハンドシェイク

SSL は、クライアントとサーバ間の接続およびサーバ間の接続を介して、ワイヤ・レベルまたはソケット・レベルで暗号化されたデータを送信する業界標準です。サーバとクライアントが一連の I/O 交換を行い、暗号化された安全なセッションをネゴシエートして合意してから、SSL 接続が確立されます。これは、「SSL ハンドシェイク」と呼ばれます。

クライアント・アプリケーションが接続を要求すると、SSL 対応サーバは証明書を提示し、ID を証明してから、データを送信します。基本的に、SSL ハンドシェイクは次の手順によって構成されています。

- クライアントはサーバに接続要求を送信します。要求には、クライアントがサポートしている SSL (または TLS: Transport Layer Security) オプションが含まれています。
- サーバは、証明書とサポートされている CipherSuite のリストを返します。これには、SSL/TLS サポート・オプション、キー交換で使用されるアルゴリズム、デジタル署名が含まれます。
- クライアントとサーバの両方が CipherSuite に合意すると、暗号化された安全なセッションが確立されます。

SSL ハンドシェイクと SSL/TLS プロトコルの詳細については、Internet Engineering Task Force Web サイト (http://www.ietf.org) を参照してください。

Open Client と Open Server がサポートしている CipherSuite のリストについては、『Open Client Client-Library/C リファレンス・マニュアル』を参照してください。

SSL のセキュリティ・レベルとセキュリティ・メカニズム

セキュリティ・レベル

Open Client と Open Server の SSL には、いくつかのセキュリティ・レベルがあります。

- SSL 対応サーバへの接続を確立すると、サーバは自己認証して接続対象のサーバであることを証明し、暗号化された SSL セッションが開始されてからデータが送信されます。
- SSL セッションが確立されると、ユーザ名とパスワードが暗号化された安全な接続によって送信されます。
- サーバ証明書のデジタル署名を比較して、サーバから受信したデータが転送中に変更されたかどうかを判断します。

セキュリティ・メカニズムとしての SSL フィルタ

SSL 対応 Adaptive Server への接続を確立すると、*sql.ini* ファイルの master 行と query 行のフィルタとして、SSL セキュリティ・メカニズムが設定されます。SSL は、TCP/IP 接続の上層に位置する Open Client/Open Server プロトコル層として使用されます。

SSL フィルタは、*sql.ini* ファイルの SECMECH (セキュリティ・メカニズム) 行で定義されている DCE や Kerberos などの他のセキュリティ・メカニズムとは異なります。master 行と query 行は、接続に使用されるセキュリティ・プロトコルを決定します。

SSL を使用する Microsoft Windows 上の一般的な *sql.ini* ファイルは、次のようになります。

```
[SERVER]
    master=TCP,hostname,address1, ssl
    query=TCP,hostname,address1, ssl
```

hostname はクライアントが接続しているサーバの名前、*address1* はホスト・マシンのポート番号です。SSL フィルタを使用して *sql.ini* ファイルの master エントリまたは query エントリに接続するには、その接続で SSL プロトコルをサポートしている必要があります。サーバを、SSL 接続を受け入れ、他の接続によってプレイン・テキスト (非暗号化データ) を受け入れるように設定したり、他のセキュリティ・メカニズムを使用するように設定できます。

たとえば、SSL ベースの接続とプレイン・テキストの接続の両方をサポートする Adaptive Server の *sql.ini* ファイルは、次のようになります。

```
[SYBSRV1]
    master=NLWNSCK,hostname,2748,ssl
    query=NLWNSCK,hostname,2748,ssl
    master=NLWNSCK,hostname,2749
    query=NLWNSCK,hostname,2749
```

この例では、SSL セキュリティ・サービスはポート番号 2748 に指定されています。SYBSRV1 では、Adaptive Server はポート番号 2749 でクリア・テキストを受信します。これには、セキュリティ・メカニズムやセキュリティ・フィルタがありません。

証明書によるサーバの有効化

Open Client/Open Server が SSL 対応サーバに接続する場合は、サーバに証明書ファイルが必要です。このファイルは、サーバの証明書と暗号化されたプライベート・キーで構成されます。また、証明書は CA がデジタル署名したものでなければなりません。

既存のクライアント接続が確立されるのと同じように、Open Client アプリケーションは Adaptive Server へのソケット接続を確立します。ネットワークのトランスポート層の接続コールがクライアント・サイドで完了し、受け入れコールがサーバ・サイドで完了すると、SSL ハンドシェイクが行われます。それから、ユーザのデータが送信されます。

SSL- 対応サーバに正しく接続するには、次の手順に従ってください。

- クライアント・アプリケーションが接続要求を行った場合は、SSL- 対応サーバは証明書を提出しなければなりません。
- クライアント・アプリケーションは、証明書に署名した CA を認識しなければなりません。「信頼された」すべての CA のリストは、信頼されたルート・ファイルにあります。詳細については、「信頼されたルート・ファイル」(89 ページ) を参照してください。
- SSL 対応サーバへの接続では、サーバ証明書の共通名は *sql.ini* ファイルのサーバ名とも一致している必要があります。

SSL- 対応 Adaptive Server への接続を確立すると、Adaptive Server は起動時に *%SYBASE%¥%SYBASE_ASE%¥certificates¥servername.crt* からサーバ自体のコード化された証明書ファイルをロードします。*servername* は、コマンド・ラインからサーバを起動したときに -S フラグで指定した Adaptive Server の名前、またはサーバの環境変数 DSLISTEN で指定した Adaptive Server の名前です。

他のタイプのサーバは、別のロケーションに証明書を保管することがあります。サーバの証明書のロケーションの詳細については、ベンダ提供マニュアルを参照してください。

SDC 環境での共通名の検証

Open Client と Open Server における SSL 検証のデフォルトの動作は、サーバ証明書での共通名を ct_connect() で指定されたサーバ名と比較することです。共有ディスク・クラスタ (SDC：Shared Disk Cluster) 環境では、クライアントはサーバ名または SDC インスタンス名とは無関係の SSL 証明書の共通名を指定できます。クライアントは、複数のサーバ・インスタンスを表すクラスタ名で SDC に接続することも、特定のサーバ・インスタンスに接続することもできます。

Open Client と Open Server は、SDC 環境での共通名の検証をサポートしています。このサポートにより、クライアントはトランスポート・アドレスを使用して、証明書の検証で使用される共通名を指定できるようになるため、Adaptive Server の SSL 証明書の共通名がサーバ名またはクラスタ名と異なっていてもかまいません。トランスポート・アドレスは、ディレクトリ・サービス (*interfaces* ファイル、LDAP、NT レジストリなど) のいずれか、または接続プロパティ CS_SERVERADDR で指定できます。

次に、Microsoft Windows での SSL 対応 Adaptive Server およびクラスタのサーバ・エントリの構文を示します。

```
[CLUSTERSSL]
query=tcp,hostname1,5000, ssl="CN=name1"
query=tcp,hostname2,5000, ssl="CN=name2"
query=tcp,hostname3,5000, ssl="CN=name3"
query=tcp,hostname4,5000, ssl="CN=name4"

[ASESSL1]
master=tcp,hostname1,5000, ssl="CN=name1"
query=tcp,hostname1,5000, ssl="CN=name1"

[ASESSL2]
master=tcp,hostname2,5000, ssl="CN=name2"
query=tcp,hostname2,5000, ssl="CN=name2"

[ASESSL3]
master=tcp,hostname3,5000, ssl="CN=name3"
query=tcp,hostname3,5000, ssl"CN=name3"

[ASESSL4]
master=tcp,hostname4,5000, ssl="CN=name4"
query=tcp,hostname4,5000, ssl="CN=name4"
```

信頼されたルート・ファイル

信頼された既知の CA のリストは、信頼されたルート・ファイルに保管されています。エンティティ (クライアント・アプリケーション、サーバ、ネットワーク・リソースなど) に既知の CA の証明書がある以外は、信頼されたルート・ファイルは証明書ファイルのフォーマットと同じです。システム・セキュリティ担当者が、標準的な ASCII テキスト・エディタを使用して CA を追加したり、削除したりします。

Open Client/Open Server の信頼されたルート・ファイルは、*%SYBASE%¥ini¥trusted.txt* にあります。

現時点で認識されている CA は、Thawte、Entrust、Baltimore、VeriSign、RSA です。

デフォルトでは、Adaptive Server はサーバ自身の信頼されたルート・ファイルを *%SYBASE%¥%SYBASE_ASE%¥certificates¥servername.txt* に格納します。

Open Client と Open Server の両方を使用すると、次のように信頼されたルート・ファイルを別のロケーションに設定できます。

- Open Client

  ```
  ct_con_props (connection, CS_SET, CS_PROP_SSL_CA,
  "SYBASEhome¥ini¥trusted.txt", CS_NULLTERM, NULL);
  ```

  ここで、*SYBASEhome* には、インストール・ディレクトリが入ります。CS_PROP_SSL_CA は、ct_config を使用してコンテキスト・レベルで設定することも、ct_con_props を使用して接続レベルで設定することもできます。

- Open Server

  ```
  srv_props (context, CS_SET, SRV_S_CERT_AUTH,
  "SYBASEhome¥ini¥trusted.txt", CS_NULLTERM, NULL);
  ```

  ここで、*SYBASEhome* には、インストール・ディレクトリが入ります。

bcp ユーティリティと isql ユーティリティでも、別の場所にある信頼されたルート・ファイルを指定できます。新しいパラメータ -x が構文に追加されており、このパラメータを使用して *trusted.txt* ファイルの場所を指定します。

証明書の取得

システム・セキュリティ担当者が、署名付きサーバ証明書とプライベート・キーをサーバにインストールします。次の手順によって、サーバ証明書を取得できます。

- ユーザ環境に展開されている既存のパブリック・キー・インフラストラクチャで提供されているサードパーティのツールを使用する。
- Sybase 証明書要求ツールをサードパーティの信頼された CA に使用する。

証明書を取得するときは、CA の証明書を要求します。サードパーティの証明書を要求し、その証明書が PKCS #12 フォーマットの場合は、certpk12 ユーティリティを使用して、証明書を Open Client/Open Server が理解できるフォーマットに変換します。詳細については、「certpk12」(98 ページ) を参照してください。

証明書要求ツールをテストし、認証方法がサーバで機能していることを確認するために、Open Client/Open Server には、certreq ツールと certauth ツールが用意されています。これらのツールを使用すると、ユーザは CA の役割を果たすことができるため、CA の署名付き証明書をユーザ自身に発行できます。

サーバで使用する証明書を作成する主な手順は、次のとおりです。

1 証明書要求を生成します。

2 パブリック・キーとプライベート・キーのペアを生成します。

3 プライベート・キーを安全な場所に保管します。

4 証明書要求を CA に送信します。

5 署名付きの証明書が CA から返信されたら、その証明書にプライベート・キーを付加します。

6 サーバのインストール・ディレクトリに証明書を保管します。

サードパーティ・ツールを使用した証明書の取得

サードパーティのほとんどの PKI ベンダと一部のブラウザでは、証明書とプライベート・キーを生成するユーティリティが用意されています。これらのユーティリティの多くはグラフィカルなウィザード形式で、一連の質問にユーザが答えると証明書の識別名と共通名が定義されます。

ウィザードの指示に従って、証明書要求を作成します。PKCS #12 フォーマットの署名付き証明書を受け取ったら、certpk12 を使用して、証明書ファイルとプライベート・キー・ファイルを生成します。この 2 つのファイルを *servername.crt* ファイルに連結します。*servername* はサーバの名前です。このファイルは、サーバのインストール・ディレクトリに配置されます。デフォルトでは、Adaptive Server の証明書は *%SYBASE%¥%SYBASE_ASE%¥certificates* に格納されます。詳細については、「certpk12」(98 ページ) を参照してください。

Sybase ツールによる証明書の要求と認証

Sybase では、証明書の要求と認可を行うためのツールを提供しています。これらのツールは、*%SYBASE%¥%SYBASE_OCS%¥bin* ディレクトリにあります。certreq は、パブリック・キーとプライベート・キーのペア、および証明書要求を生成します。certauth は、サーバ証明書要求を CA の署名付き証明書に変換します。

警告！ certauth は、テストだけを目的として使用してください。商用 CA のサービスを利用することをおすすめします。こうしたサービスでは、ルート証明書の整合性が保護されているためです。また、広く承認された CA によって署名された証明書を使用することで、クライアント証明書を使用した認証への移行が容易になります。

次の手順 1 ～ 5 に従って、サーバの信頼されたルート証明書を用意します。サーバ証明書を作成できることを確認するために、5 つの手順をすべて行い、テスト版の信頼されたルート証明書を作成します。テスト版の CA 証明書 (信頼されたルート証明書) がある場合は、手順 3 ～ 5 を繰り返してサーバ証明書に署名します。

❖ サーバの信頼されたルート証明書を用意する

1　certreq を使用して、証明書を要求します。

2　certauth を使用して、証明書要求を CA の自己署名付き証明書 (信頼されたルート証明書) に変換します。

3　certreq を使用して、サーバ証明書とプライベート・キーを要求します。

4　certauth を使用して、証明書要求を CA 署名付きサーバ証明書に変換します。

5　プライベート・キーのテキストをサーバ証明書に付加して、サーバのインストール・ディレクトリに証明書を格納します。

注意 certauth と certreq は、RSA と DSA のアルゴリズムに依存しています。これらのツールは、RSA および DSA の各アルゴリズムを使用して証明書要求を構築する、ベンダ提供の暗号モジュールでのみ動作します。

以下のリファレンスの項では、前の手順で使用したツールについて説明します。

Adaptive Server でサーバ証明書を追加、削除、表示する方法については、『ASE システム管理ガイド』を参照してください。

certauth

サーバ証明書要求を CA (認証機関) の署名付き証明書に変換します。

構文

certauth
[-r]
[-C *caCert_file*]
[-Q *request_filename*]
[-K *caKey_filename*]
[-N *serial_number*]
[-O *SignedCert_filename*]
[-P *caPassword*]
[-s *start_time*]
[-T *valid_time*]
[-v]

パラメータ

-r

テスト環境用の自己署名付きルート証明書を作成します。

-C *caCert_file*

-r を指定した場合は、CA の証明書要求ファイルの名前を指定します。それ以外の場合は、CA のルート証明書の名前を指定します。

-Q *request_filename*

証明書要求ファイルの名前を指定します。

-K *caKey_filename*

CA のプライベート・キーの名前を指定します。

-N *serial_number*

署名付き証明書のシリアル番号を指定します。-N が指定されていない場合、certauth は疑似ランダム・シリアル番号を生成します。

-O *SignedCert_filename*

署名付き証明書ファイルを作成する場合に出力用に使用する名前を指定します。-r が指定されている場合、*SignedCert_filename* は自己署名付きルート証明書になります。-r オプションを使用しない場合、*SignedCert_filename* は *caCert_file* によって署名された証明書です。

-P *caPassword*

プライベート・キーの復号化に使用する CA のパスワードを指定します。

-s *start_time*

証明書の有効期間の開始時刻を指定します。有効期間は日単位で計算されます。-s を指定しなかった場合は、現在の時刻がデフォルトの開始時刻となります。

-T *valid_time*

証明書の有効期間を指定します。有効期間は日単位で計算されます。

-v

certauth ツールのバージョン番号と版権メッセージを表示して、終了します。

例 1

この例では、プライベート・キー (*ca_pkey.txt*) を使用して、CA の証明書要求 (*ca_req.txt)* を証明書に変換します。プライベート・キーは *password* で保護されています。この例では、有効期間を 365 日に設定し、証明書に自己署名してルート証明書 (*trusted.txt*) として出力します。

```
certauth -r -C ca_req.txt -Q ca_req.txt
-K ca_pkey.txt -P password -T 365 -O trusted.txt
```

ユーティリティは、次のメッセージを返します。

```
 -- Sybase Test Certificate Authority --
Certificate Validity:
    startDate = Tue Sep 5  10:34:43  2000

    endDate = Wed Sep 5  10:34:43  2001

CA sign certificate SUCCEED (0)
```

注意　テスト CA 用に trusted ルート証明書を 1 回だけ作成する必要があります。信頼されたルート証明書を作成したら、この証明書を使用して、テスト環境の多くのサーバ証明書に署名できます。

例 2

この例では、サーバ証明書要求 (*srv5_req.txt*) を証明書に変換し、有効期間を 180 日に設定します。ここでは、CA の証明書 (*trusted.txt*) とプライベート・キー (*ca_pkey.txt*) を持つ証明書に署名し、パスワード保護を使用し、署名付き証明書を *sybase_srv5.crt* として出力します。

```
certauth -C trusted.txt -Q srv5_req.txt
-K ca_pkey.txt -P password -T 180 -O sybase_srv5.crt
```

注意　有効期間を設定しない場合は、デフォルトの 365 日が使用されます。

ユーティリティは、次のメッセージを返します。

```
 -- Sybase Test Certificate Authority --
Certificate Validity:
    startDate = Tue Sep  5 10:38:32  2000

    endDate = Sun Mar  4 09:38:32  2001

CA sign certificate SUCCEED (0)
```

次に、証明書の例を示します。サーバが使用できるサーバ証明書の作成手順については、次の「使用法」の項を参照してください。

```
-----BEGIN CERTIFICATE-----

MIICSTCCAgUCAVAwCwYHKoZIzjgEAwUAMG8xCzAJBgNVBAYTAlVTMRMwEQYDVQQI
EwpDYWxpZm9ybmlhMRMwEQYDVQQHEwpFbWVyeXZpbGxlMQ8wDQYDVQQKFAZTeWh
c2UxDDAKBgNVBAsUA0RTVDEXMBUGA1UEAxQOc3liYXNlX3Rlc3RfY2EwHhcNMDAw
ODE4MTkxMzM0WhcNMDEwODE4MTkxMzM0WjBvMQswCQYDVQQGEwJVUzETMBEGAUE
CBMKQ2FsaWZvcm5pYTETMBEGA1UEBxMKRW1lcnl2aWxsZTEPMA0GA1UEChQGU3li
YXNlMQwwCgYDVQQLFANEU1QxFzAVBgNVBAMUDnN5YmFzZV90ZXN0X2NhMIHwMIo
BgcqhkjOOAQBMIGcAkEA+6xG7XCxiklxbP96nHBnQrTLTCjHlcy8QhIekwv9OlqG
EMG9AjJLxj6VCkPOD75vqVMEkaPPjoIbXEJEe/aYXQIVAPyvY1+B9phC2e2YFcf7
cReCcSNxAkBHt7rnOJZ1Dnd8iLQGt0wd1w4lo/Xx2OeZS4CJW0KVKkGId1hNGz8r
GrQTspWcwTh2rNGbXxlNXhAV5g4OCgrYA0MAAkA70uNEl90Kmhdt3RISiceCMgOf
1J8dgtWF15mcHeS8OmF9s/vqPAR5NkaVk7LJK6kk7QvXUBY+8LMOugpJf/TYMAsG
AhUAhM2Icn1pSavQtXFzXJUCoOmNLpkCFQDtE8RUGuo8ZdxnQtPu9uJDmoBiUQ==

-----END CERTIFICATE-----
```

使用法

- -N オプションで指定するシリアル番号の最大長は、16 進文字で 20 文字です。指定したシリアル番号がこれよりも長い場合、certauth はシリアル番号を最大長にトランケートします。
- Adaptive Server が認識するサーバ証明書ファイルを作成するには、署名付き証明書ファイルの最後に証明書リクエスタのプライベート・キーを追加します。上記の例のように、*srv5_pkey.txt* を切り取って、署名付き証明書ファイル *sybase_srv5.crt* の最後に貼り付けます。
- サーバが起動時にロードできる信頼されたルート・ファイルを作成するには、ファイル名 *trusted.txt* を *sybase_srv5.txt* に変更します。*sybase_srv5.txt* はサーバの共通名です。
- 次に、*sybase_srv5.txt* ファイルを Adaptive Server インストール・ディレクトリにコピーします。たとえば、*%SYBASE%¥%SYBASE_ASE%¥certificates* にコピーします。

このファイルは、SSL ベースのセッションに必要であり、SSL 対応の Adaptive Server を起動するときに使用されます。

CA のルート証明書を作成したら、この証明書を使用して、複数のサーバ証明書に署名できます。

参照

certreq

certreq

サーバ証明書要求と対応するプライベート・キーを作成します。このユーティリティは対話型モードで使用できます。また、コマンド・ラインにオプションのパラメータをすべて提供できます。

構文

certreq
[-F *input_file*]
[-R *request_filename*]
[-K *PK_filename*]
[-P *password*]
[-v]

パラメータ

-F *input_file*

属性情報のある入力ファイル名を指定して、証明書要求を構築します。*input_file* 名を指定しない場合は、必要な情報をユーザが対話形式で入力します。

input_file には、次のエントリが必要です。

```
req_certtype={Server,Client}
req_keytype={RSA,DSA}
req_keylength={for RSA: 512-2048;
     for DSA: 512,768,1024}
req_country={string}
req_state={string}
req_locality={string}
req_organization={string}
req_orgunit={string}
req_commonname={string}
```

注意 複数のサーバが同じ共通名を使用できるクラスタ環境以外では、共通名はサーバ名と同じ名前にしてください。

詳細については、「SDC 環境での共通名の検証」(88 ページ) を参照してください。

サンプル・ファイルの *input_file* については、例 2 を参照してください。

-R *request_filename*

証明書要求ファイルの名前を指定します。

-K *PK_filename*

プライベート・キー・ファイルの名前を指定します。

-P *password*

プライベート・キーを保護するために使用されるパスワードを指定します。

-v

バージョン番号と版権メッセージを表示して、終了します。

例 1

この例では、-F *input_file* パラメータを使用していないため、対話型モードになります。サーバ証明書要求 (*server_req.txt*) とプライベート・キー (*server_pkey.txt*) を作成するには、次のように入力します。

```
certreq

Choose certificate request type:
     S - Server certificate request
     C - Client certificate request (not supported)
     Q - Quit
Enter your request [Q] : s

Choose key type:

     R - RSA key pair
     D - DSA/DHE key pair
     Q - Quit
Enter your request [Q] : r

Enter key length (512, 768, 1024 for DSA; 512-2048 for RSA) :
512

Country: US

State: california

Locality: emeryville

Organization: sybase

Organizational Unit: dst

Common Name: server
```

ユーティリティから次のメッセージが返されます。

```
Generating key pair (please wait) . . .
```

キーのペアが生成されると、certreq ユーティリティは、さらに多くの情報の入力を要求するプロンプトを表示します。

```
Enter password for private key : password

Enter file path to save request: server_req.txt

Enter file path to save private key : server_pkey.txt
```

例 2

または、非対話型モードに -F オプションを使用することもできます。-F オプションを使用する場合は、有効値を使用し上記で説明したフォーマットに従ってください。これらに誤りがある場合、証明書は正しく作成されません。

次は、認証要求の非対話型エントリに使用できるサンプル・テキスト・ファイルです。

```
certreq -F input_file

req_certtype=server
req_keytype=RSA
req_keylength=512
req_country=us
req_state=california
req_locality=emeryville
req_organization=sybase
req_orgunit=dst
req_commonname=server
```

このファイルを作成、保存してから、コマンド・ラインに次のように入力します。

```
certreq -F path_and_file -R server_req.txt
-K server_pkey.txt -P password
```

ここでは、*path_and_file* には、テキスト・ファイルのロケーションが入ります。

このファイルは、サーバ証明書要求 (*server_req.txt*) とプライベート・キー (*server_pkey.txt*) を作成するものです。プライベート・キーは、*password* で保護されます。

サーバ証明書ファイルは、標準的な ASCII テキスト・エディタを使用して編集できます。

使用法

- 入力ファイルでは、<tag>=*value* のフォーマットを使用します。<tag> では大文字と小文字が区別されるため、上記のとおりに入力してください。
- “=” は必須です。有効な *value* は、文字または数字で始まり、単一のワードであることが必要です。また、*value* の中にスペースを含めないでください。
- *value* が必要な <tag> は、“req_certtype”、“req_keytype”、“req_keylength”、“req_commonname” です。
- <tag>、“=”、*value* の前後のスペースまたはタブは許容されます。ブランク行も許容されます。
- 各コメント行は、# で始めてください。
- 証明書要求ファイルは、PKCS #10 フォーマットになっています。この証明書要求ファイルは、certauth ツールが要求を CA の署名付き証明書に変換するときに受け入れ可能な入力として使用されます。

参照

certauth

certpk12

PKCS #12 ファイルを証明書ファイルとプライベート・キーにエクスポートまたはインポートします。

構文

certpk12
{*-O Pkcs12_file* | *-I Pkcs12_file*}
[-C *Cert_file*]
[-K *Key_file*]
[-P *key_password*]
[-E *Pkcs12_password*]
[-v]

パラメータ

-C *Cert_file*

-O がオンの場合は、PKCS #12 ファイルにエクスポートする証明書ファイルの名前を指定します。-I がオンの場合は、PKCS #12 ファイルからインポートする証明書ファイルの名前を指定します。

-K *Key_file*

-O がオンの場合は PKCS #12 ファイルにエクスポートするプライベート・キー・ファイルの名前、または -I がオンの場合は PKCS #12 ファイルからインポートするプライベート・キー・ファイルの名前を指定します。

-P *Key_password*

-K を指定しているプライベート・キーの保護に使用するパスワードを指定します。-O がオンの場合は、プライベート・キーを PKCS #12 ファイルにエクスポートするためのパスワードが必要です。-I がオンの場合は、PKCS #12 ファイルからプライベート・キーをインポートした後にテキスト・ファイルに出力するためのパスワードが必要です。

-O *Pkcs12_file*

エクスポートする PKCS #12 ファイルの名前を指定します。ファイルの内容は、証明書とプライベート・キー、証明書だけ、プライベート・キーだけの 3 つの場合があります。-O または -I のどちらかがオンになっていなければなりません。

-I *Pkcs12_file*

インポートする PKCS #12 ファイルの名前を指定します。ファイルの内容は、証明書とプライベート・キー、証明書だけ、プライベート・キーだけの 3 つの場合があります。-I または -O のどちらかがオンになっていなければなりません。

-E *Pkcs12_password*

PKCS #12 ファイルを保護するために使用するパスワードを指定します。-O がオンの場合は、エクスポートする PKCS #12 ファイルを暗号化するときにパスワードを使用します。-I がオンの場合は、インポートする PKCS #12 ファイルを復号化するときにパスワードを使用します。パスワードは「トランスポート・パスワード」とも呼ばれます。

-v

certpk12 ツールのバージョン番号と版権メッセージを表示して、終了します。

例 1

この例では、証明書ファイル (*caRSA.crt*) とプライベート・キー・ファイル (*caRSApkey.txt*) を PKCS #12 ファイル (*caRSA.p12*) にエクスポートします。*password* は、*caRSApkey.txt* の復号化に使用するパスワードです。*pk12password* は、最後の *caRSA.p12* の暗号化に使用するパスワードです。

```
certpk12 -O caRSA.p12 -C caRSA.crt -K caRSApkey.txt
     -P password -E pk12password

-- Sybase PKCS #12 Conversion Utility certpk12 Thu Nov 9
16:55:51 2009--
```

例 2

この例では、証明書とプライベート・キーを含む PKCS #12 ファイル (*caRSA.p12*) をインポートします。埋め込み証明書をテキスト・ファイル "*caRSA_new.crt*" に出力し、埋め込みプライベート・キーをテキスト・ファイル "*caRSApkey_new.txt*" に出力します。*new_password* は、*caRSApkey_new.txt* を保護するために使用し、*pk12password* は *caRSA.p12* ファイルの復号化に必要です。

```
certpk12 -I caRSA.p12 -C caRSA_new.crt
  -K caRSApkey_new.txt  -P new_password
-E pk12password

-- Sybase PKCS#12 Conversion Utility certpk12 Thu Nov 9
16:55:51 2009--
```

注意　例 1 と 2 を実行すると、*caRSA.crt* と *caRSA_new.crt* は同じ内容になります。*caRSApkey.txt* と *caRSApkey_new.txt* はランダムに暗号化されるため、内容が異なります。

例 3

この例では、証明書ファイル (*caRSA.crt*) を PKCS#12 ファイル (*caRSAcert.p12*) にエクスポートします。*pkcs12password* は、*caRSAcert.p12* の暗号化に使用します。

```
certpk12 -O caRSAcert.p12 -C caRSA.crt
-E pk12password

-- Sybase PKCS#12 Conversion Utility certpk12 Thu Nov 9
16:55:51 2009--
```

例 4

この例では、証明書を含む PKCS #12 ファイル (*caRSAcert.p12*) をインポートします。埋め込み証明書をテキスト・ファイル (*caRSAcert.txt*) に出力します。*pk12password* は、*caRSAcert.p12* ファイルの復号化に必要です。

```
certpk12 -I caRSAcert.p12 -C caRSAcert.txt
    -E pk12password

-- Sybase PKCS#12 Conversion Utility certpk12 Thu Nov 9
16:55:51 2009--
```

注意　例 3 と例 4 を実行すると、*caRSA.crt* と *caRSAcert.txt* は同じ内容になります。

使用法

- certpk12 がサポートしているのは、トリプル DES 暗号化方式で暗号化された PKCS #12 ファイルだけです。
- 証明書要求者のプライベート・キーを署名付き証明書ファイルの最後に付加します。
- ファイルに *servername*.crt という名前を付けます。*servername* はサーバの名前です。これを *%SYBASE%¥%SYBASE_ASE%* の下の証明書ディレクトリに配置します。

  このファイルは、SSL 対応の Adaptive Server を起動するときに必要です。

参照

certreq と certauth

カスタマイズされた Open SSL のサポート

Microsoft Windows x86-64 64 ビット版では、Open SSL を使用して SSL 機能をサポートします。

SSL 機能を有効にするには、*libsybfcsissl64.dll* ランタイム・ライブラリを *libtcl64.cfg* 設定ファイルに追加します。設定ファイルは、*%SYBASE%¥%SYBASE_OCS%¥ini* にあります。

パスワード暗号化のための FIPS 140-2 準拠

Open Client と Open Server のログイン・パスワードとリモート・パスワードの暗号化は、Sybase CSI (Common Security Infrastructure) によって実現されます。Certicom SSL Plus 5.2.2 CSI-Crypto 2.6 は、連邦情報処理標準 (FIPS: Federal Information Processing Standard) 140-2 に準拠しています。FIPS 暗号化をサポートするために、SDK または Open Server のインストール時に、*sbgse2.dll* という名前の Certicom Security Builder 共有ライブラリが *%SYBASE%/%SYBASE_OCS%/lib3p* または *%SYBASE%/%SYBASE_OCS%/lib3p64* にインストールされます。

索引

N

O

P

S

U

か

き

け

さ

し

せ

そ

て

と

ね

は

ふ

へ

ほ

ろ