SYBASE®

設定ガイド

Open Client™ and Open Server™

12.5.1

Microsoft Windows 版

ドキュメント ID：DC35837-01-1251-01
改訂：2003 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

この『Open Client/Server 設定ガイド Windows 版』では、次のプラットフォームでシステムを設定して Open/Client Server™ 製品を実行する方法について説明します。

- Microsoft Windows NT
- Microsoft Windows 2000
- Microsoft Windows 2003
- Microsoft Windows XP

このマニュアルでは、特に明記しないかぎり、4 つの Microsoft Windows プラットフォームすべてを「Windows」と表記します。

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よりも、システム管理の点から、設定作業とトピックを説明します。

このマニュアルの内容

『Open Client/Server 設定ガイド Windows 版』は次の 3 部によって構成されています。

- 設定手順
- 設定ユーティリティ
- 設定に関する参照情報

設定手順　最初の 6 つの章では、特定の設定作業を実行するための手順ごとの説明など、設定の各部分について説明します。

- 「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 CyberSafe Kerberos の両方のセキュリティ・サービスに関する情報も提供します。

設定ユーティリティ　以下の章では、設定作業に使用できるユーティリティについて説明します。

- 「第 7 章 ocscfg の使い方」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 と wocscfg 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。

設定に関する参照情報　付録では、設定手順に関する章に列挙している作業のリファレンス情報について説明しています。設定トピックは、設定情報のソースごとの付録として分類されています。

- 「付録 A 環境変数」では、Open Client/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、次の点について説明します。
  - *libtcl.cfg* (ドライバ設定ファイル)
  - *sql.ini* (interfaces 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、次の点について説明します。
  - *locales.dat* ファイル
  - *objectid.dat* ファイル
  - *mnemonics.dat* ファイル
  - 変換設定ファイル
  - 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 サポートと、SSL (Security Socket Layer) プロトコルの使用に必要なシステム設定作業について要約します。

関連マニュアル

- 『リリース・ノート Open Client/Server』では、リリースについての最新情報を提供しています。
- 『Sybase 製品インストール・ガイド』では、Open Client/Server ソフトウェアをインストールするため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』では、Open Client Client-Library™ のリファレンス情報について説明しています。
- 『Open Client DB-Library/C リファレンス・マニュアル』では、DB-Library™ についての参照情報を提供します。
- 『Open Client Client-Library/C プログラマーズ・ガイド』では、Client-Library プログラムの設計方法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』では、Open Server Server-Library のリファレンス情報について説明しています。
- 『Open Client/Server Common Libraries』では、CS-Library のリファレンス情報について説明しています。CS-Library は、Client-Library と Server-Library の両方のアプリケーションで役に立つユーティリティ・ルーチンの集まりです。
- 『Open Client/Server プログラマーズ・ガイド補足』では、Open Client/Server 製品を使用するプログラマのために、プラットフォーム固有の情報について説明しています。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/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、およびシステム・プロシージャについて説明しています。
- 『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拡張版である Sybase の Transact-SQL® について説明しています。このマニュアルでは、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に、テキストブック形式で説明しています。

その他の情報ソース

Sybase Getting Started CD、Sybase Technical Library CD、Technical Library 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・ガイドが収録されています。また、その他のマニュアルや、Technical Library CD には含まれない更新情報が収録されることもあります。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)。
- Technical Library CD には製品マニュアルが入っており、この CD は製品のソフトウェアに同梱されています。DynaText リーダー (Technical Library CD に収録) を使用すると、この製品に関する技術情報に簡単にアクセスできます。

  Technical Library のインストールと起動の方法については、マニュアル・パッケージに含まれている『Technical Library Installation Guide』を参照してください。
- Technical Library Product Manuals Web サイトは、Technical Library CD の HTML バージョンで、標準の Web ブラウザを使っ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。

  Technical Library Product Manuals Web サイトにアクセスするには、Product Manuals (http://www.sybase.com/support/manuals/) にアクセスしてください。

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. 左側のナビゲーション・バーから [Products] を選択します。
3. 製品リストから製品名を選択し、[Go] をクリックします。
4. [Certification Report] フィルタを選択し、時間枠を指定して [Go] をクリックします。
5. [Certification Report] のタイトルをクリックして、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 Page (http://www.sybase.com/support) を指定します。

2 [EBFs/Maintenance] を選択します。すでに Web アカウントをお持ちの場合はユーザ名とパスワードを要求されますので、各情報を入力します。Web アカウントをお持ちでない場合は、新しいアカウントを作成します。サービスは無料です。

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。

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。

表記の規則

表 1: 構文の表記規則

構文要素	意味
command	コマンド名、コマンド・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などのキーワードは、**太字**で表記される。
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。

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。このマニュアルを読む前に、SDK の『インストール・ガイド』の指示に従って、Open Client か Open Server またはその両方をインストールしてください。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を説明します。この章の内容は、次のとおりです。

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) と Net-Library™ を提供します。これらを使用することで、Adaptive Server Enterprise、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・パーティの製品、その他の Sybase 製品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 リファレンス・マニュアル』
- 『Open Client DB-Library リファレンス・マニュアル』
- 『Open Server Server-Library 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。
- SQL Server®、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。

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、セキュリティ・ドライバ、ネットワーク (Net-Library) 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お互いに通信します。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なければなりません。

❖ **Open Client から接続を確立する**

1 DSQUERY 環境変数を使用して、ターゲット・サーバ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、ターゲット・サーバのアドレスを取得します。

注意 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かぎって、Open Client は DSQUERY を使用します。

❖ 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　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かぎって、Open Client は DSLISTEN を使用します。

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を行ってください。

- 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 と Open Server 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 に必要な基本の設定を説明します。この章の内容は、次のとおりです。

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
- ネットワーク・ドライバの名前
- 使用するセキュリティ・メカニズム

SYBASE 環境変数に定義されている 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 10.0.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は、コマンド・プロンプトで SYBASE 環境変数と PATH 環境変数を設定する必要があります。

次の手順では、バージョン 12.5 以降の製品を使用するアプリケーションが *C:¥SYBASE* にインストールされます。

1 コマンド・プロンプトを開き、12.5.1 ディレクトリ用に SYBASE 環境変数と PATH 環境変数を設定します。次に例を示します。

```
set SYBASE=C:¥SYBASE¥
set SYBASE_OCS=OCS-12_5
set PATH=%PATH%;¥%SYBASE%¥%SYBASE_OCS%¥bin;C:
¥%SYBASE%¥%SYBASE_OCS%¥dll
```

2 別のコマンド・プロンプトを開き、12.5.1 ディレクトリ用に SYBASE 環境変数と PATH 環境変数を設定します。次に例を示します。

```
set SYBASE=C:¥SYBASE
set SYBASE_OCS=OCS-12_5
set PATH=%PATH%;C:¥%SYBASE%¥%SYBASE_OCS%¥bin;C:
¥%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、sLanguage が定義されていない場合は、Open Client はロケール名として「デフォルト」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こ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 (すべての Microsoft Windows プラットフォーム用) 内のエントリを探して、サーバ・アドレス情報をどこで調べるかを決定します。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を決定します。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、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、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、それに対応するターゲット・アドレスを使用します。

  バージョン 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 -P -Usa -Sconnect50 -Ic:¥sybase
```

ネットワーク・ドライバの名前

(DB-Library には適用されません) Open Client は、Windows の *libtcl.cfg* ファイルの [DRIVERS] セクションを調べて、どのネットワーク・ドライバをロードするかを決定します。

注意 使用するプロトコルに対する Net-Library ドライバのみをインストールしてください。それ以外の場合は、エラー・メッセージが表示されます。

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、Windows の *libtcl.cfg* の [SECURITY] セクションを調べて、どのセキュリティ・ドライバを使用するかを決定します。

Adaptive Server バージョン 12.5 以降では、以前のバージョンで格納されたデータとは異なる制限を持つデータを格納できます。Open Client バージョン 12.5 以降は、Adaptive Server 12.5.1 の制限をサポートします。12.5 より前のバージョンの Open Client と Open Server を使用している場合は、次の処理を行ったときにデータを処理できません。

- Adaptive Server バージョン 12.5 以降へのアップグレード
- 幅広いカラムを持つテーブルの削除と再作成
- 長いデータの挿入

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

❖ **環境変数の設定**

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「デフォルト」エントリがアプリケーションで使用するローカライゼーション値を反映していることを確認してください。

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、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 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って環境変数を設定する方法については、「環境変数の設定」(41 ページ) を参照してください。

ドライバの設定

ネットワーク、ディレクトリ、セキュリティの各ドライバを設定するには、ocscfg を使用します。

ドライバの設定方法については、「第 7 章 ocscfg の使い方」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」(66 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 項目が 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てください。

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」(72 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 に必要な基本の設定を説明します。この章の内容は、次のとおりです。

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 環境変数に定義されている Sybase インストール・ディレクトリのロケーション

リリース 10.0.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は、コマンド・プロンプトで SYBASE 環境変数と PATH 環境変数を明示的に設定する必要があります。

次の手順に従って、*c:¥SYBASE* ディレクトリにある 12.5 以降の製品を使用するアプリケーションをインストールします。

❖ **SYBASE 環境変数と PATH 環境変数の設定**

1 コマンド・プロンプトを開き、12.5.1 ディレクトリ用に SYBASE 環境変数と PATH 環境変数を設定します。次に例を示します。

```
set SYBASE=C:¥SYBASE
set SYBASE_OCS=OCS-12_5
set PATH=%PATH%;C:¥%SYBASE%¥%SYBASE_OCS%¥bin;C:¥%SYBASE%¥
%SYBASE_OCS%¥dll
```

2 別のコマンド・プロンプトを開き、12.5.1 ディレクトリ用に SYBASE 環境変数と PATH 環境変数を設定します。次に例を示します。

```
set SYBASE=C:¥SYBASE
set SYBASE_OCS=OCS-12_5
set PATH=%PATH%;C:¥%SYBASE%¥%SYBASE_OCS%¥bin;C:¥%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は、Open Server は「**デフォルト**」をロケール名として使用します。

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、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・アドレス情報をどこで調べるかを決定します。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、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、それに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 -P -Usa -Sconnect50 -Ic:¥sybase¥ini¥sql.ini
  ```

注意　使用するプロトコルに対する Net-Library ドライバのみをインストールしてください。それ以外の場合は、エラー・メッセージが表示されます。

ネットワークベースのセキュリティ・サービスを使用する接続をクライアントが要求している場合は、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

各タスクについて、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* またはレジストリを設定する**

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 項目が 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てください。

dsedit の使い方については、「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」(72 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「デフォルト」エントリがアプリケーションで使用するローカライゼーション値を反映していることを確認してください。

  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、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って環境変数を設定する方法については、「環境変数の設定」(41 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 の使い方」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」(66 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の詳細については、『高可用性システムにおける Sybase フェールオーバの使用』を参照してください。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 HA フェールオーバをサポートしていません。C 版と 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。

この章の内容は、次のとおりです。

interfaces 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・プロパティを設定するには、*sql.ini* ファイルに “hafailover” というラベルの行を追加し、クライアントがセカンダリ・コンパニオンに接続するのに必要な情報を提供します。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。

“MONEY1” と “PERSONNEL1” コンパニオンを対照型に設定する場合の *sql.ini* のエントリを次に示します。

```
[MONEY1]
 query=TCP,FN1,9835
 master=TCP,FN1,9835
 hafailover=PERSONNEL1
[PERSONNEL1]
 query=TCP,HUM1,7586
 master=TCP,HUM1,7586
 hafailover=MONEY1
```

interfaces ファイルにこの情報を追加する方法については、使用しているプラットフォームの『Open Client/Server 設定ガイド』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は、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できる必要があります。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 のフェールオーバ・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 と ct_con_props 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。次のコードを使用して、コンテキストまたは接続レベルのどちらかにこのプロパティを設定できます。

```
CS_INT true = CS_TRUE;
CS_INT false = CS_FALSE;
retcode = ct_config(context, CS_SET, CS_HAFAILOVER, &true, CS_UNUSED,
NULL);
retcode = ct_con_props(connection, CS_SET, CS_HAFAILOVER, &false,
CS_UNUSED, NULL);
```

2　フェールオーバ・メッセージの処理コンパニオンが落ち始めると、クライアントはフェールオーバが発生しようとしているという情報メッセージを受信します。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、interfaces 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、フェールオーバが設定されていても、interfaces ファイルに hafailover サーバのエントリが無い場合 (またはその逆) 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 を発行するまでは、フェールオーバ・プロパティ (たとえば、isql -Q など) にクライアントを接続することはできません。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。

- interfaces 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。

interfaces ファイルに、「interfaces 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。この章の内容は、次のとおりです。

注意 DB-Library はディレクトリ・サービスをサポートしていません。

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。

使用できるディレクトリ・サービス・プロバイダは、プラットフォームごとに異なります。次の表は、各プラットフォームに有効なディレクトリ・サービス・プロバイダのリストです。

プラットフォーム	Windows レジストリ	Novell NDS	LDAP
Microsoft Windows NT	x		x
Microsoft Windows 2000	x		x
Microsoft Windows 2003	x		x
Microsoft Windows XP	x		x

LDAP

Lightweight Directory Access Protocol (LDAP) 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。

LDAP サーバは一般的には階層構造で、高速なリソースの検索が可能です。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。

実際のサーバやほかの LDAP サービスへのゲートウェイかどうかにかかわらず、どのタイプの LDAP サーバでも 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 interfaces ファイルの比較

LDAP ディレクトリ・サービスは、通常の Sybase interfaces ファイル (Windows では *sql.ini*) の代わりとなるものです。interfaces ファイルでは、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、interfaces 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、中央リポジトリにユーザ、リソース、セキュリティ情報を統合することにより、リソース情報の管理が大幅に簡略化されます。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。役割は、ユーザがアクセスできる情報とサーバ、および保持している読み込み／書き込みパーミッションがある場合は、そのパーミッションを定義します。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。

次の表は、Sybase interfaces ファイルと LDAP サーバの相違点を示します。

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

interfaces ファイル	ディレクトリ・サービス
プラットフォーム固有	各プラットフォームに共通
各 Sybase インストール固有	集中階層型
別のマスタ・エントリとクエリ・エントリを含む	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できる

従来の interfaces ファイルは、TCP 接続の UNIX マシンおよびフェールオーバ・マシンで次のように表示されます。

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

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

次の例は、TCP 接続の LDAP エントリとフェールオーバ・マシンを示します。

```
dn: sybaseServername=foobar, dc=sybase,dc=com
objectClass: sybaseServer
sybaseVersion: 12500
sybaseServername: foobar
sybaseService: ASE
sybaseStatus: 4
sybaseAddress: TCP#1#foobar 5000
sybaseRetryCount: 12
sybaseRetryDelay: 30
sybaseHAServernam: secondary
```

LDAP ディレクトリ・サービスへ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設定します。表 5-2 は、有効な DIT ベースの値を示します。

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 または Sybase SQL Server。
sybaseStatus	整数	ステータス。1 = アクティブ、2 = 停止、3 = 失敗、4 = 不明。

属性名	値のタイプ	説明
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。次の項目を含む。 • プロトコル：TCP、NAMEDPIPE、SPX DECNET (大文字と小文字を区別して入力する) • 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 注意 dscp は、この属性をトランスポート・タイプとトランスポート・アドレスに分割します。
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。

Windows での Sybase の LDAP ディレクトリ・スキーマのリストは、*%SYBASE%¥%SYBASE_OCS%¥ini* にあります。同じディレクトリに、*sybase-schema.conf* と呼ばれるファイルがあります。このファイルには、同じスキーマですが、Netscape 固有の構文のものがあります。

上記の例では、エンティティはポート番号 5000 の TCP 接続を受信する “foobar” という名前の Adaptive Server を表しています。このエンティティには、12 (回) というリトライ回数と 30 (秒) というリトライ待ち時間も指定されています。sybaseRetryCount と sybaseRetryDelay は、それぞれ CS_RETRY_COUNT と CS_LOOP_DELAY にマップされています。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” など) 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。表 5-2 (23 ページ) の sybaseAddress を参照してください。

次の例は、異なる接続プロトコルの 2 つのアドレスで受信している NT サーバの LDAP エントリを示します。

```
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。表 5-2 (23 ページ) はアドレス属性の各フィールドの値の定義を示します。

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、変更できます。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検索します。

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」(66 ページ) を参照してください。

アプリケーションでのディレクトリ・サービスの使い方

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

1 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_S_DSPROVIDER) を呼び出している場合) は、*libtcl.cfg* の DIRECTORY セクションを検証し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。

   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」(66 ページ) を参照してください。

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は、Client-Library と Server-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 次のいずれかが当てはまる場合は、Client-Library と Server-Library は *interfaces* ファイルに戻り、そこから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は、*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、パスワードを指定し、LDAP サーバへの接続を認証します。

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() によって設定されます。その特定の接続に ct_con_props() 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* ファイルのパスワードまたはユーザ名のどちらかを指定しない場合は、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は、パスワードは暗号化されていると想定されます。詳細については、「パスワードの暗号化」(68 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。

アプリケーションでの LDAP ディレクトリ・サービスの使い方

Sybase LDAP の機能を使用するには、ベンダ提供マニュアルに従って、LDAP サーバをインストールして設定します。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。これは、*%SYBASE%¥%SYBASE_OCS%¥dll* にあります。

Netscape LDAP SDK ライブラリのロケーションと環境変数は、表 5-4 (29 ページ) にリストされています。

警告！ Sybase LDAP ディレクトリ・サービスでは、DB-Library で構築されたクライアント・アプリケーションはサポートしていません。

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という 2 つの認証方法をベースとした接続を確立します。

- 匿名アクセス － 認証情報を必要としないので、属性を設定する必要はありません。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 (「libtcl.cfg ファイルと libtcl64.cfg ファイル」(66 ページ) を参照) の拡張機能として *libtcl.cfg* ファイル (64 ビットのプラットフォームでは、*libtcl64.cfg* ファイル) で指定するか、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、安全でないこともありますが、その場合はユーザ名とパスワードの認証を使用できます。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

- 接続の確立には、パスワードなどの認証情報は必要ありません。
- 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 属性で設定することもできます。

libtcl.cfg* ファイルと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、再入可能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 ベンダ提供の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、次のように修正します。

- *libtcl*.cfg* ファイルの *[DIRECTORY]* エントリにある LDAP URL 行の行頭から、セミコロン (;) のコメント・マーカを削除します。
- *[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。

警告！ LDAP URL は、1 行で記述する必要があります。

```
ldap=libdldap.so ldap://host:port/ditbase??scope????
     bindname=username password
```

次に例を示します。

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

“*one*” は、DIT ベースの 1 つ下のレベルのエントリを取り出す検索のスコープを示します。表 5-3 は、*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_* property
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　正しい環境変数が、必要なサードパーティ・ライブラリを指していることを確認してください。表 5-4 は、LDAP SDK ライブラリのロケーションのリストです。

表 5-4: 環境変数

プラットフォーム	環境変数	ライブラリのロケーション
Windows NT、2000	PATH	*%SYBASE%¥%SYBASE_OCS¥lib3p*
Windows 2003、XP	PATH	*%SYBASE%¥%SYBASE_OCS¥lib3p*

5　dscp または dsedit を使用して、LDAP サーバに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」(51 ページ) と「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・サービスのすべてが LDAP サーバである必要はありません。次に例を示します。

```
[DIRECTORY]
   ldap=libdldap.so ldap://test:389/dc=sybase,dc=com
   dce=libddce.so ditbase=/.:/subsys/sybase/dataservers
   ldap=libdldap.so ldap://huey:11389/dc=sybase,dc=com
```

この例では、*test:389* への接続が失敗すると、指定された DIT ベースを持つ DCE ドライバへの接続にフェールオーバします。この接続も失敗すると、*huey:11389* 上の LDAP サーバに接続しようとします。ベンダが異なると、DIT ベースのフォーマットも異なります。これらの構造体については、『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 の使い方については、「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 の使い方」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」(66 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どのように機能するかと、この機能を使用するにはどのような設定が必要かを説明します。この章の内容は、次のとおりです。

注意 ネットワークベースのセキュリティは Client-Library 11.1 以降で機能します。このため、Open Server リリース 11.1 以降をベースにしたサーバが必要です。Open Client DB-Library と SQL Server 11.0 はネットワークベースのセキュリティ・サービスをサポートしません。

ネットワークベース・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操作したりするおそれがあります。この問題に対処するために、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、接続用のセキュリティ・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。異なるプラットフォームは異なるセキュリティ・メカニズムを使用できます。

LAN Manager Windows NT 版 (SSPI) と CyberSafe Kerberos は、Windows NT 3.51 以降のサーバと Windows NT (3.51 以降) と Windows 2000 のクライアントにセキュリティ・サービスを提供しています。

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

- オプションの *sql.ini* のエントリの 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。
- オプションのディレクトリ・サービス・エントリの secmech 属性には、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するセキュリティ・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は、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 行や属性がない場合は、すべてのセキュリティ・メカニズムを使用できます。
- secmech 行または属性があってもセキュリティ・メカニズムがリストされていない場合、サーバはセキュリティ・メカニズムをサポートしません。

セキュリティ・ドライバ

Client-Library および Server-Library がセキュリティ・メカニズムと通信することを可能にするセキュリティ・ドライバを、Sybase では提供しています。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。

接続上でセキュリティ・メカニズムを使用するには、次の 2 つの条件がどちらも満たされていなければなりません。

- クライアントとサーバは、互換性のあるセキュリティ・ドライバを使用しなければなりません。たとえば、Windows LAN Manager ドライバを使用するクライアントには、Windows LAN Manager ドライバを使用するサーバが必要です。
- 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することによってサービスを要求する必要があります。

注意 ネットワークベースのセキュリティには、リリース 11.1 以降の Open Server で構築されたサーバが必要です。SQL Server 11.0 では、Open Server 11.1 以降のゲートウェイが使用されていないかぎり、ネットワークベースのセキュリティをサポートしません。

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安全な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参照して、どのサービスが実行されているかを決定します。

LAN Manager セキュリティ・サービス

Windows LAN Manager サービスは、Windows NT 版 Open Server、Windows NT、Windows 2000 に次の機能を提供します。

- 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

CyberSafe 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。

CyberSafe Kerberos セキュリティ・サービス

CyberSafe 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
- データの機密保持
- リプレイの検出
- 順序不整合の検出

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方法の概要については、「Client-Library とセキュリティ・サービス」(36 ページ) を参照してください。

注意 ここで説明する作業の中には、CyberSafe 管理ツールが必要なものもあります。詳細については、CyberSafe 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。

- 次のように、CyberSafe ソフトウェアをシステムにインストールします。Open Client/Open Server 12.5 以降の場合は CyberSafe Challenger 5.3.1 以降、Open Client/Open Server 12.5 以降の場合は CyberSafe TrustBroker をインストールします。
- CyberSafe Application Development Kit バージョン 1.1 を使用します。
- ct_con_props を使用してクレデンシャル (希望のセキュリティ機能) を設定したり、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。
- *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。
- アプリケーションは、サーバに接続するのに、すでに作成されているユーザ・クレデンシャルを使用しなければなりません。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 にログインする必要があります。
- CyberSafe ユーティリティ kinit を使用して CyberSafe セキュリティ・メカニズムにログインし、Client-Library 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、その名前はそのユーザ・クレデンシャルと一致していなければなりません。ユーザ名を入力しないと、Client-Library はそのユーザの CyberSafe 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。
- これらの環境変数では、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

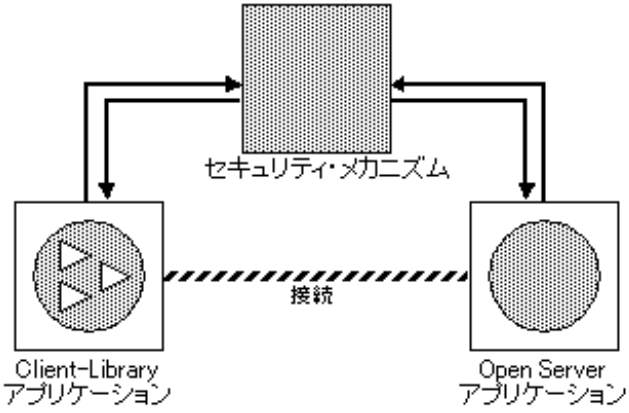
  詳細については、CyberSafe のマニュアルを参照してください。

- Client-Library アプリケーションの実行中は、*gssapi.dll* と *csfc516.dll* の 2 つの DLL ファイルがパスに含まれていなければなりません。これら 2 つの DLL は Sybase によっては提供されていませんが、CyberSafe 製品には含まれています。これら 2 つの DLL が CyberSafe 製品に含まれていない場合は、CyberSafe に連絡して GSS-API ライブラリを入手します。
- 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 を設定したら、isql を使用して設定を検査できます。isql の [ファイル] - [セキュリティ] がグレー表示されている場合は、セキュリティ・サービスは利用できません。Open Client、Open Server、CyberSafe が正しく設定されているかどうかを確認してください。

アプリケーションでのセキュリティ・サービスの使い方

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。ログインが有効の場合は、Open Server アプリケーションは安全な接続を確立します。

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サインを作成します。

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暗号解読と検証を行います。

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 は接続に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は、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参照して、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ています。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、システム管理者はサポートするセキュリティ・メカニズムを制限できます。

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」(87 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、ocscfg32 ユーティリティを使用してセキュリティ・ドライバを設定してください。セキュリティ・ドライバの設定方法については、「第 7 章 ocscfg の使い方」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」(66 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 の使い方

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。この章の内容は、次のとおりです。

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 の使い方については、「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 4 つのタイプの情報を設定できます。

- 環境変数
- Net-Library ドライバ
- 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、[ファイル マネージャ] のどこからでも起動できます。Windows 2000 や NT 4.0 の場合は、[スタート] メニューから、または Windows 98 の [エクスプローラ] や NT 4.0 の [Windows NT 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・グループの ocscfg アイコンをダブルクリックします。
- DOS 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 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。

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 は、起動時にデフォルトでこのダイアログ・ボックスを選択する。
Net-Library	*libtcl.cfg* ファイルにリストされている Net-Library ドライバを設定する。
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してください。

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と、ocscfg は *libtcl.cfg* ファイルを検索できません。

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。

2　選択した名前が [Variable Name] ボックスに表示されます。

3　[Clear] をクリックします。

Net-Library ドライバの設定

[Net-Library] タブをクリックして、Net-Library 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 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。

Net-Library ドライバの追加または変更

❖ **Net-Library ドライバを追加または変更する**

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 [Protocol] フィールドで、設定するドライバのネットワーク・プロトコルを選択します。

3 [Net-Library] ボックスから、設定するドライバを選択します。

ドライバ	説明
NLWNSCK	Winsock TCP/IP ドライバ
NLMSNMP	Named Pipes ドライバ
NLNWLINK	SPX/IPX ドライバ
NLDECNET	DECnet ドライバ

注意 変更はただちに有効になります。

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・エントリの追加**

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」(43 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、Registry DIT ベース設定の例を 2 つ示します。 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です。 すべての DIT ベース・エントリは、*¥HKEY_LOCAL_MACHINE¥* を起点にしていなければなりません。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・エントリが必要です。 *¥HKEY_LOCAL_MACHINE¥¥SOFTWARE¥SYBASE* キーは、Sybase インストール・プログラムが作成します。上記の例の場合は、ユーザが *SERVER* キーを追加する必要があります。 Microsoft regedt32 ツールを使用して、必要なキーを作成します。Registry パス名では、大文字と小文字を区別しません。
Novell NDS	CN=*server*.OU=*organization_unit*.O=*organization*

DIT ベースを指定しなかった場合は、ディレクトリ・ドライバは表 7-2 のデフォルト値を使用します。

表 7-2: デフォルト DIT ベース値

ディレクトリ・サービス	DIT ベース構文
Windows レジストリ	SOFTWARE¥SYBASE¥SERVER
Novell NDS	ユーザの Novell セッションからの名前コンテキスト

既存のディレクトリ・ドライバ・エントリの変更

❖ **既存のディレクトリ・ドライバ・エントリの変更**

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、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。
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. [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。
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. [Save 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・エントリの追加**

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. [Add] 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」(87 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] ボタンをクリックします。

既存のセキュリティ・ドライバ・エントリの変更

❖ **既存のセキュリティ・ドライバ・エントリの変更**

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・サービスや *sql.ini* を設定する方法を説明します。この章の内容は、次のとおりです。

dsedit の使用

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* 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 引数を指定しない場合は、dsedit は最初のダイアログ・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] から dsedit を起動するには、次の手順に従ってください。

  a　*%SYBASE%¥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と、dsedit は *libtcl.cfg* を探すことができません。

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」(54 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、汎用ツールにはないセマンティック・チェックが組み込まれている dscp または dsedit を使用することをおすすめします。

各サーバ・エントリは、一連の属性で構成されます。サーバ・エントリを追加または変更すると、dscp から、サーバ属性に関する情報の入力を指示するプロンプト画面が表示されます。属性のいくつかはデフォルトで提供されますが、そのほかはユーザが入力する必要があります。dscp を使用してディレクトリ・サービスを作成すると、角カッコ “[ ]” 内にデフォルト値が表示されます。入力可能な値のリストについては、表 5-2 (23 ページ) を参照してください。

注意 LDAP に対して dscp を使用する場合は、*libtcl.cfg* ディレクトリの LDAP エントリの後にスペースを入力できません。

dsedit は、*libtcl*.cfg* ファイルと interfaces (Windows NT 上の *sql.ini*) ファイル内のサーバ・エントリを追加、削除、変更するためのグラフィカル・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、変更を行ってください。詳細については、「libtcl.cfg ファイルと libtcl64.cfg ファイル」(66 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

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

1 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%¥%SYBASE_OCS%¥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、次の URL を Netscape に入力します。

```
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 Version	整数	サーバ・オブジェクト定義のバージョン・レベル。 オブジェクト定義の将来の変更を識別するために、Sybase がこの属性を提供する。	110
Server Name	文字列	サーバ名。	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。	SQL Server
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
Security Mechanism	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ための、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

属性名	値のタイプ	説明	デフォルト値
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 (2 フォーマット) 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 (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* • DECnet (4 つのフォーマットがある) 1. *area_number.node_number,object_name* 2. *area_number.node_number,object_number* 3. *node_name,object_name* 4. *node_name,object_number*	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、エントリを変更してください。

サーバ・エントリの変更

❖ **サーバ・エントリの変更**

1. [Server] ボックスで、サーバ・エントリをクリックします。
2. [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。各属性の説明は、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**

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」(58 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、*_n* というバージョン番号の付いたサーバ・エントリのコピーを付加します。[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」(53 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」(53 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 SQL Server、Adaptive Server、または Open Server アプリケーションの間のネットワーク接続を検査する方法を説明します。この章の内容は、次のとおりです。

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、SQL Server、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、SQL 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。

問題のタイプによっては、dsedit では診断できません。通常は、Net-Library とネットワーク・ソフトウェア間の接続ではなく、SQL Server、Adaptive Server、または Open Server の設定に関わる問題がこれに該当します。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」(59 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、次の基本的な Net-Library の稼働条件がすべて満たされているかどうかを確かめます。

- SQL Server、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 PC がハードウェアとソフトウェアの最低稼働条件を満たしている。
- ネットワーク・ベンダのソフトウェアがユーザ 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。

警告！ Net-Library DLL が 1 コピーだけ PC にインストールされていることを確認してください。

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を知らせてきた場合、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。

注意　接続情報が正しくない場合は (たとえば、dsedit が名前付きパイプを使用しようとしたが、ユーザのネットワーク・ソフトウェアは TCP/IP プロトコルを使用している場合など)、dsedit は別のネットワーク用にインストールされている Net-Library バージョンの *sql.ini* ファイルでサーバ情報を探すこともあります。テストに適したサーバを選択していることを確認してください。

---

dsedit は、Net-Library DLL をロードできない場合でも、そのことを知らせます。"Unable to Load" メッセージが表示された場合は、PATH 環境変数に指定されているディレクトリに Net-Library DLL が存在すること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」(60 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 はエラーを報告しなかったが、他のアプリケーションが実行できなかった場合には、次の手順に従ってください。

- 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。
- SQL Server、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。
- 使用する Net-Library ドライバが *libtcl.cfg* ファイルにリストされていることを確認します。
- isql ユーティリティを使用して、SQL Server、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。

- SQL 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。SQL Server、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 が入っているファイル・リストを表示します。

一般的な質問

- 質問：Net-Library Driver バージョン 11.x は ODBC をサポートしますか。

  答え：サポートしません。ただし、バージョン 1.x の Net-Library ではサポートします。

- 質問：新しいバージョンの Sybase DLL を入手したのに、ソフトウェアはまだ旧バージョンの動作をします。

  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、Net-Library ドライバ 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：*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

この章では、設定情報となる環境変数を説明します。この章の内容は、次のとおりです。

接続に使用する環境変数

Open Client/Open Server 製品は、接続処理時に表 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は、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は、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-12_5	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 製品は、ローカリゼーション時に表 A-2 に示されている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/Open 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。

付 録　B

設定ファイル

この章では、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を説明します。この章の内容は、次のとおりです。

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	関連項目
libtcl.cfg	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・ドライバに関する情報と必要な初期化情報が入っている。	*SYBASE_home¥OCS-12_5¥ini*	「libtcl.cfg ファイルと libtcl64.cfg ファイル」(66 ページ) 参照。
sql.ini	interfaces 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入っている。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_home¥ini*	「sql.ini ファイル」(72 ページ) 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_home¥ini*	「付録 C ローカライゼーション」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_home¥ini*	「ocs.cfg」(76 ページ) 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 (*libtcl*.cfg* ファイル) は、Open Client/Open Server 製品で使用する 3 つのタイプのドライバ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
- ネットワーク (Net-Library) 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。たとえば、Open Client は Winsock TCP/IP Net-Library ドライバ *NLWNSCK* を使用して、Winsock TCP/IP プロトコルを使用するサーバに接続できます。

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* は両方とも、64 ビット・プラットフォーム上で提供されます。設定情報を探す場合は、dsedit や srvbuild などの (64 ビット・プラットフォームの) 32 ビット・アプリケーションでは *libtcl.cfg* ファイル、64 ビット・アプリケーションでは *libtcl64.cfg* ファイルを検索します。

libtcl.cfg* ファイルには、interfaces ファイルや LDAP ディレクトリ・サービスを使用するかどうかを指定します。*libtcl*.cfg* ファイルに LDAP が指定してある場合は、サーバ接続時に -I パラメータを渡すことによってアプリケーションが明示的に *libtcl*.cfg* ファイルを上書きしないかぎり、interfaces ファイルは無視されます。

libtcl.cfg は *SYBASE_home¥OCS-12_5¥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分かれた複数のセクションによって構成されています。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	Windows ディレクトリ・ドライバのリスト
SECURITY	Windows セキュリティ・ドライバのリスト
DRIVERS	Windows ネットワーク (Net-Library) ドライバのリスト

注意 セクションの順序は任意です。

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
```

各パラメータの意味は、次のとおりです。

- *provider* には、ディレクトリ・サービスのローカル名が入ります。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* には、ドライバの名前が入ります。ドライバのデフォルト・ロケーションは *SYBASE_home¥OCS-12_5¥dll* です。*driver* のオプションは次のとおりです。

ドライバ名	説明	使用対象
LIBDREG	Windows レジストリ・ドライバ	Windows NT、2000
WDSNDS	Novell NDS ドライバ	Windows
WDSBAN	Banyan Street Talk ドライバ	Windows

- *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

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¥* を起点にしている必要がある。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。

Novell NDS	CN=*server*.OU=*organization_unit*.O=*organization*

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 ベース情報にこ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、LDAP ディレクトリ・サービスは、次のようなフォーマットによって指定されています。

```
[DIRECTORY]
ldap=libdldap.so 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可能な場合にだけ動作します。

```
ldap=libdldap.so ldap://test:389/dc=sybase,dc=com
```

libtcl.cfg* ファイルでユーザ名とパスワードを LDAP URL への拡張機能として指定すると、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 と *libtcl64.cfg* ファイルのエントリは人間の目で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、キーボード入力を行うと、パスワードと置換される暗号値を生成する単純なアルゴリズムです。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　要求されたら、パスワードを 2 度入力します。pwdcrypt が、次のように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。

```
ldap=libdldap.so
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=libdldap.so
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た場合でも、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
```

各パラメータの意味は、次のとおりです。

- *provider* には、セキュリティ・メカニズムのローカル名が入ります。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_home¥locales¥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」(87 ページ)を参照してください。

- *driver* には、ドライバの名前が入ります。ドライバのデフォルト・ロケーションは *SYBASE_home¥OCS-12_5¥dll* です。*driver* のオプションは次のとおりです。

ドライバ名	説明	使用対象
LIBSMSSP	Windows LAN Manager ドライバ	Windows NT、2000
WSSMSSP	Windows LAN Manager ドライバ	Windows (クライアントのみ)
WSSKRB	CyberSafe Kerberos セキュリティ・ドライバ	Windows (クライアントのみ)

- *init_string* (CyberSafe Kerberos のみ) 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。

  CyberSafe Kerberos ドライバの詳細については、*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。

  token=value

  有効なトークンは次のとおりです。

  - secbase=*@location*。location は CyberSafe Kerberos がプリンシパルがあると期待しているロケーションです。

CyberSafe セキュリティ・サービス初期化構文

CyberSafe セキュリティ・ドライバのサポートが Open Client/Open Server に追加されています。CyberSafe 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* ディレクトリを編集します。

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・ボックスに入力します。

- *Local Name*：*csfkrb5* または *objectid.dat* ファイルに CyberSafe 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 LIBSKRB 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、CyberSafe セキュリティ・ドライバの *provider* 値を *csfkrb5* または *objectid.dat* ファイルの CyberSafe 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 LIBSKRB 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 CyberSafe プリンシパルを持つレルムです。Windows NT と Windows 95 では、このエントリが必要です。たとえば、次のようになります。

```
[SECURITY]
csfkrb5=LIBSKRB 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この構文を使用します。

secbase=/*...dce_cell_name*

次に例を示します。

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

Net-Library ドライバ

Net-Library ドライバ・エントリの構文は次のとおりです。

```
driver=protocol description
```

各パラメータの意味は、次のとおりです。

- *driver* には、ドライバの名前が入ります。ドライバのデフォルト・ロケーションは *%SYBASE%¥%SYBASE_OCS%¥dll* です。*driver* のオプションは次のとおりです。

ドライバ	説明
NLWNSCK	**Winsock TCP/IP ドライバ**
NLMSNMP	Named Pipes ドライバ
NLNWLINK	SPX/IPX ドライバ
NLDECNET (NT のみ)	DECnet ドライバ

- *protocol* はネットワーク・プロトコルの名前で、ドライバのプロトコルと一致していなければなりません。有効な値は次のとおりです。
  - TCP/IP の場合、“TCP”
  - Named Pipe の場合、“NAMEPIPE”
  - SPX/IPX の場合、“SPX”
  - DECnet の場合、“DECNET”
- *description* はドライバのオプション説明です。

libtcl.cfg の例

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


[DRIVERS]
 NLWNSCK=TCP  Winsock TCP/IP Net-Lib driver
 NLMSNMP=NAMEPIPE  Named Pipe Net-Lib driver
```

```
NLNWLINK=SPX  NT NWLINK SPX/IPX Net-Lib driver
NLDECNET=DECNET  DecNET Net-Lib driver


[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 の使い方」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・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します。
- Open Server は *sql.ini* ファイル・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します。

Open Client/Open Server 製品が *sql.ini* を探すのに、アプリケーションは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」と『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 の使い方については、「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・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
```

各パラメータの意味は、次のとおりです。

- *SERVERNAME* は Open Client または Open Server がどの *sql.ini* エントリを読み込むのかを認識するときに使用するエイリアスです。*SERVERNAME* には、アルファベット (ASCII a-z、A-Z) で始まり、アルファベット、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。

  Windows NT と 2000 では、*service_type* のオプションは次のとおりです。

  - master 行には、“master”。これはサーバ・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します。
  - query 行には、“query”。これはクライアント・アプリケーションがサーバを探すときに使用します。

  sql.ini エントリの master 行と query 行には、同じ情報が入っています。dsedit は各エントリに両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・ドライバのリストの詳細については、「Net-Library ドライバ」(71 ページ) を参照してください。
- *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
DECnet	次の 4 種類がある。 1. *area_number.node_number,object_name* 2. *area_number.node_number,object_number* 3. *node_name,object_name* 4. *node_name,object_number*	1.23,SQLSERVER1 1.23,214 COLLIN,OBJECT_222 COLLIN,214

- “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。カンマ (“,”) をセパレータとして使用して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」(87 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* の例をリストします。

プロトコル	プラットフォーム	例
TCP/IP	Windows NT Windows 2000	[SYBASE] master=NLWNSCK,TEST,8877 query=NLWNSCK,TEST,8877 secmech=1.3.6.1.4.1.897.4.6.3
名前付きパイプ	Windows NT Windows 2000	[SYBASE] master=NLMSNMP,¥PIPE¥SQL¥`QUERY  query=NLMSNMP,¥¥TEST¥PIPE¥SQL¥`UERY secmech=1.3.6.1.4.1.897.4.6.3
IPX/SPX	Windows NT Windows 2000	[SYBASE] master=NLNWLINK,TEST query=NLNWLINK,TEST secmech=1.3.6.1.4.1.897.4.6.3
DECnet	Windows NT Windows 2000	[SYBASE] master=NLDECNET,1.23,SQLSERVER1 query=NLDECNET,1.23,SQLSERVER1 secmech=1.3.6.1.4.1.897.4.6.3

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* を解析して、サーバ名 MYSERVER を見つけると、このサーバは受信 TCP/IP 接続では TCP/IP アドレス “mercury,1234” で受信し、受信 IPX/SPX 接続では 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” でサーバに接続しようとします。この接続に失敗すると、“plato,9876” でサーバに接続しようとします。この接続に失敗すると、名前付きパイプ・プロトコルを使用して、“¥¥plato¥pipe¥sql¥query” でサーバに接続しようとします。これにも失敗すると、最後に、IPX/SPX プロトコルを使用して、“my_mercury_spx” でサーバに接続します。この最後の接続にも失敗すると、Open Client はエラーを返します。

ocs.cfg

Client-Library アプリケーションは、*ocs.cfg* ランタイム設定ファイルを使用して、次の設定をします。

- 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点です。

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 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。

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。

この章では、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 の『開発者用国際化ガイド』を参照してください。この付録の内容は、次のとおりです。

ローカライゼーション・プロセスの概要

Open Client/Open Server アプリケーションのローカライズには次の 2 つの方法があります。

- 初期ローカライゼーション値を使用する
- 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を使用する。

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の実行中、特定の時点でローカライズする必要があった場合、Open Client/Open Server アプリケーションはカスタム・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

アプリケーションがカスタム・ローカライゼーション時にどの環境変数を使用するかについては、Open Client/Open Server の『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。
  - ローカライズしたメッセージ・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 つの言語と 1 つ以上の文字セットおよび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

実行時に、Open Client/Open Server アプリケーションは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。Sybase リリース・ディレクトリの 3 つのディレクトリに、これらのファイルが入っています。

- *locales* ディレクトリは次のディレクトリとファイルから構成されます。
  - 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。
  - すべての製品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。このディレクトリは言語名別に編成されています。
  - 以前のリリースの Open Client/Open 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。
- *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。それぞれの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。
  - *objectid.dat* ファイル。これは、オブジェクトのグローバル識別子をローカルのプラットフォーム固有の名前にマップします。
  - *mnemonic.dat* ファイル。必要に応じてソース Unicode と置換するためのニモニック文字列が含まれています。

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 または Open Server アプリケーションを設定する場合は、上記のディレクトリに、ユーザ・サイトとユーザ・アプリケーションに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 専用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 はサーバへの送信時には iso_1 をクライアント文字セットとしてデフォルトで使用します。iso_1 を使用する場合には、文字セットを変更してデータ破損を防ぎます。次のいずれかの方法でこれを行います。

- *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 オプションを付けて 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、どのローカライゼーション情報をロードするかを決定します。*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入っていません。

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に定義されたロケール定義エントリが入っています。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は「デフォルト」ロケールを定義するエントリが指定されています。

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]
```

各パラメータの意味は、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* ファイルの末尾にあるコメントに、ロケール名の POSIX 値がリストされています。
- *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ので、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、エントリがすでにあるかどうかも確認してください。

次の処理ができます。

- 「デフォルト」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せます。たとえば、次のように Sybase であらかじめ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は LC_ALL 環境変数に “french” の値が必要な場合は、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* に使用する値を選択します。

2 *language_name* の値を決定します。

  Sybase 言語モジュールがインストールされると、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 の言語モジュールがインストールされると、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それぞれの文字セット用の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うちの 1 つ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を選択する場合は、*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*) があります。*sortorder_name* は、*.srt* 部分を除いて、これらのファイル名のうちの 1 つと一致している必要があります。

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、次のタスクを行ってください。

- 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すでに追加していて、既存のアプリケーションが 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て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、そのエントリを使用するアプリケーションが 1 つもないことを確認してください。

ローカライズしたメッセージ・ファイル

警告！ ローカライズしたメッセージ・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入っています。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) を使用して、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ます。

すべての Open Client/Open 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入っ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する場合、インストール処理で *language_name* サブディレクトリが新規に作成され、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、たいていは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

すべての Open Client/Open Server 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、サポートする各文字セットの変換ファイルと照合順ファイルが入っています。

変換設定ファイル

各文字セットの変換設定ファイルには、変換プロセスに関する情報が入っています。

変換設定ファイルの使い方

クライアントとサーバが異なる文字セットを使用する場合は、文字セット間での変換が必要となります。Open Client/Open Server 製品には、各文字セットの変換をサポートするファイルが含まれています。

文字セットの変換設定ファイルには、変換に使用するモードとマップ不可能な文字に使用する置換文字が指定されています。

表 C-1 は、変換モードの説明です。

表 C-1: コード化文字セット変換のモード

モード	説明
MATCH 出荷時に提供されているファイルにはこの値が入っている。	変換処理では、変換元と変換先とで一致している値が変換される。 変換元の文字のコードが正しくないか、またはマップ不可能である場合は、変換処理では、変換先の文字セットの変換設定ファイルに定義されている送信先置換文字を使用する。
BESTGUESS	変換処理では、変換元と変換先とで一致している値と近似の値が変換される。 変換元の文字のコードが正しくないか、またはマップ不可能である場合は、変換処理では、変換先の文字セットの変換設定ファイルに定義されている送信先置換文字を使用する。
MNEMONIC	変換元と変換先とで一致している値が変換される。変換元の値に対して一致している値がない場合、変換処理では変換先の値として Unicode のニモニック文字列が使用される。適切なニモニック文字列がない場合は、変換処理では変換先の値として Unicode の 16 進数文字列が使用される。 変換元の文字のコードが正しくない場合は、変換処理では、変換先の文字セットの変換設定ファイルに定義されている送信先置換文字を使用する。

文字セット変換処理の詳細については、Open Client/Open Server の『開発者用国際化ガイド』を参照してください。

変換設定ファイルのロケーション

各文字セットに、変換設定ファイルがあります。このファイルは *Sybase_home¥charsets¥charset_name¥charset_name.cfg* にあります。

Sybase_home¥charsets ディレクトリ構造図の詳細については、「ローカライゼーション・ファイル」(79 ページ) を参照してください。

変換設定ファイル・エントリ

変換セクションには、特定の文字セットへの変換がどのように行われるかを記述するエントリが入っています。変換セクションのエントリは、テーブル駆動型の変換かアルゴリズム駆動型の変換かを示す場合があります。

テーブル駆動型の変換の場合、エントリの形式は次のようになります。

```
[conversion]
     convertto = dest_charset, table, mode, replacement_char
```

各パラメータの意味は、次のとおりです。

- *dest_charset* は変換先の文字セットの名前です。
- カンマ (,) ファイルのリスト・セパレータ文字です。
- table は変換がテーブル駆動型であることを示すキーワードです。
- *mode* は使用する変換モードです。テーブル駆動型の変換だけに適用されます。有効な値は次のとおりです。
  - MATCH
  - BESTGUESS
  - MNEMONIC

  各モードの詳細については、表 C-1 を参照してください。
- *replacement_character* は MATCH と BESTGUESS モード変換時に使用する送信先置換文字の 16 進 (“0x” プレフィクスなしの) コードです。

アルゴリズム駆動型の変換の場合、エントリの形式は次のようになります。

```
[conversion]
     convertto = dest_charset, sys_algorithm, multiplier
```

各パラメータの意味は、次のとおりです。

- *dest_charset* は変換先の文字セットの名前です。
- カンマ (,) ファイルのリスト・セパレータ文字です。
- sys_algorithm は変換処理が標準 Open Client/Open Server 変換アルゴリズムを使用することを示すキーワードです。
- *multiplier* は変換に使用する変換乗数を表す整数値です。この値は、変換時の文字列長の増加量の最大値を示しています。

変換設定ファイルの例

次に、変換設定ファイルの例を示します。

```
; Conversion config File for iso_1 charset.
 [conversion]
 convertto = utf8, table, MATCH, 3F
 convertto = cp850, sys-algorithm, 1
 convertto = cp437, sys-algorithm, 1
 convertto = roman8, sys-algorithm, 1
 convertto = mac, sys-algorithm, 1
```

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%¥locales¥message* 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りますが、一般に次のようなファイルです。

- *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 の『開発者用国際化ガイド』を参照してください。

ini ディレクトリ

ini ディレクトリには、次のファイルが入っています。

- グローバル・オブジェクト識別子ファイル (*objectid.dat*)
- ニモニック・ファイル (*mnemonic.dat*)

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
     1.3.6.1.4.1.897.4.6.2 = nds, novellsecmech
     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* を編集します。

mnemonic.dat ファイル

mnemonic.dat には、文字セット変換中に必要に応じて、マップ不可能な変換元の文字を置き換えるために使用できる POSIX ニモニック文字列が入っています。

mnimonic.dat には、UCS-2 <-> ニモニック文字列変換だけが入っています。出荷時の *mnemonic.dat* ファイル内にある各 Unicode ニモニックは、Unicode 文字を表す XPG4 文字の文字列です。

mnemonic.dat は POSIX ニモニック文字列を Unicode UCS-2 文字コードと対応させて機能します。このニモニック文字列は XPG4 Portable Character Set の文字を使用しているので、どの変換先の文字セットに使用するにも適しています。

mnemonic.dat の使い方

変換先の文字セットの変換設定ファイル (*charset.cfg*) で “Mnemonic” モードが指定されているときだけ、*mnemonic.dat* が使用されます。この場合、変換時に *mnemonic.dat* が次のように使用されます。

1 変換元の文字が変換先の文字セットにはマップ不可能であるとわかった場合、Sybase ソフトウェアは変換元の文字を Unicode UCS-2 に変換します。

2 Sybase は *mnemonic.dat* ファイルの UCS-2 コードを調べ、送信先データ・ストリームで、これに対応するニモニック文字列を使用します。

3 *mnemonic.dat* ファイルに適切な文字列が含まれていない場合は、送信先データ・ストリームには Unicode UCS-2 16 進文字列を使用します。

文字セット変換処理の詳細については、Open Client/Open Server の『開発者用国際化ガイド』を参照してください。

mnemonic.dat のロケーション

mnemonic.dat は *%SYBASE%¥charsets* ディレクトリにあります。

Sybase_home¥charsets ディレクトリ構造図の詳細については、「ローカライゼーション・ファイル」(79 ページ) を参照してください。

mnemonic.dat のエントリ

mnemonic.dat には、UCS-2 コードをニモニック文字列と対応させるエントリが入っています。

ニモニック・セクションのエントリの形式は次のようになります。

```
mnem = <mnem_string> <UCS-2_encoding> comment
```

各パラメータの意味は、次のとおりです。

- < は無視されます。
- *mnem_string* はニモニック文字列を表す XPG4 の文字列です。
- > はファイルのリスト・セパレータ文字です。
- *UCS-2_encoding* は文字の UCS-2 コードです。
- *comment* はコメント文字列です。

ニモニック・セクションのエントリは、その他の Sybase ローカライゼーション・ファイルのエントリとは多少異なります。これは、*mnemonic.dat* ファイルでは標準 POSIX 定義を使用しているためです。

mnemonic.dat の例

次に、ニモニック・ファイルの例を示します。

```
[file format]
      version = 11.0
      escape = /
      list_separator = >

[copyright]

     copyright = "Copyright ... ."

[mnemonics]
      mnem = <NU> <U0000> NULL (NUL)
      mnem = <SH> <U0001> START OF HEADINGS(SOH)
      mnem = <SX> <U0002> START OF TEXT (STX)
      mnem = <EX> <U0003> END OF TEXT (ETX)
      mnem = <ET> <U0004> END OF TRANSMISSION (EOT)
      mnem = <EQ> <U0005> ENQUIRY (ENQ)
      mnem = <AK> <U0006> ACKNOWLEDGE (ACK)
```

```
mnem = <BL> <U0007> BELL (BEL)
mnem = <BS> <U0008> BACKSPACE (BS)
.
.
.
```

mnemonic.dat への文字列の追加

Sybase が出荷する *mnemonic.dat* には、すべての文字セットのすべての文字のニモニック文字列が含まれているわけではありません。必要な文字列が *mnemonic.dat* に含まれていない場合は、vi などのオペレーティング・システムのエディタを使用して文字列を追加できます。

ftp サイトの *unicode.org* には、既存の文字列の更新だけでなく、新しい Unicode ニモニック文字列についての情報もあります。

付 録　D

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

この章では、Open Client/Open Server の SSL サポートと、SSL プロトコルの使用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。

この付録の内容は、次のとおりです。

SSL ハンドシェイク

SSL は、クライアントからサーバ、およびサーバからサーバへワイヤまたはソケット・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は何度か I/O を交換し、安全な暗号化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」と呼ばれています。

クライアント・アプリケーションが接続を要求すると、SSL 対応サーバが証明書を提示し、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 CipherSuites 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ています。
- クライアントとサーバがお互いに CipherSuite に合意すると、安全で暗号化された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/Open Server がサポートしている CipherSuites の詳細については、『Open Client Client-Library 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、サーバは接続対象のサーバであることを自己認証し、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安全で暗号化された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、*interfaces* ファイル (Windows では *sql.ini*) の master 行と query 行のフィルタに SSL セキュリティ・メカニズムが設定されます。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 SSL を使用します。

SSL フィルタは、*interfaces* ファイル (Windows では *sql.ini*) の、SECMECH (セキュリティ・メカニズム) 行で定義される 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 と query 行では、接続のために必要なセキュリティ・プロトコルを指定します。

たとえば、tli (トランスポート・レイヤ・インタフェース) と SSL を使用している UNIX マシンの一般的な *interfaces* ファイルは、次のようになります。

```
SERVER <retries><time-outs>

 query tli tcp /dev/tcp tli_add1 ssl
 master tli tcp /dev/tcp tli_add1 ssl
```

SSL を使用している Windows NT 上の一般的な *sql.ini* ファイルは、次のようになります。

```
[SERVER]

query=TCP,hostname,address1, ssl
master=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、*interfaces* ファイルの master エントリまたは query エントリに接続しようとする場合は必ず、SSL プロトコルをサポートする必要があります。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 UNIX の Adaptive Server の *interfaces* ファイルは、次のようになります。

```
SYBSRV1
    master tli tcp /dev/tcp ¥x00020abc1234567800000000000000000 ssl
    query tli tcp /dev/tcp ¥x00020abc1234567800000000000000000 ssl
    master tli tcp /dev/tcp ¥x00020abd1234567800000000000000000
```

または、UNIX 上の新しいスタイルの Sybase *interfaces* ファイルのある同じエントリは、次のようになっています。

```
SYBSRV1
    master tli tcp hostname 2748 ssl
    query tli tcp hostname 2748 ssl
    master tli tcp hostname 2749
```

ソケットスタイルの *interfaces* ファイルの例は、次のようになります。

```
SYBSRV1
    master tcp ether hostname 2748 ssl
    query tcp ether hostname 2748 ssl
    master tcp ether hostname 2749
```

これらの例では、SSL セキュリティ・サービスはポート番号 2748 (0x0abc) に設定されています。SYBSRV1 では、Adaptive Server はポート番号 2749 (0x0abd) 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、サーバは証明書ファイルが必要です。これには、サーバの証明書と暗号化プライベート・キーが含まれています。また、証明書は認証局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、証明書に署名した認証局を認識しなければなりません。「信頼済み」認証局すべてを含んだリストは、信頼済みルート・ファイルにあります。詳細については、「信頼済みルート・ファイル」(94 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 interfaces ファイルのサーバ名とも一致していなければなりません。

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 の名前か、またはサーバの環境変数 $DSLISTEN で指定した Adaptive Server の名前です。

ほかのタイプのサーバでは、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。

信頼済みルート・ファイル

既知で信頼済みの認証局のリストは、信頼済みルート・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) に既知で認証局の証明書がある以外は、信頼済みルート・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 ASCII テキスト・エディタを使って認証局を追加したり、削除したりします。

Open Client/Open Server の信頼済みルート・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認証局は、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済みルート・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、次のように信頼済みルート・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,
"$SYBASE/config/trusted.txt", CS_NULLTERM, NULL);
```

$SYBASE はインストール・ディレクトリです。ct_config() を使用してコンテキスト・レベルに、または ct_con_props() を使用して接続レベルに CS_PROP_SSL_CA を設定できます。

- Open Server

```
srv_props (context, CS_SET, SRV_S_CERT_AUTH,
"$SYBASE/config/trusted.txt", CS_NULLTERM, NULL);
```

$SYBASE はインストール・ディレクトリです。

証明書の取得

システム・セキュリティ担当者が、署名済み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済み認証局に使用する。

証明書を取得するには、認証局に証明書を要求してください。サードパーティに証明書を要求し、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、Open Client/Open Server が理解できるフォーマットに証明書を変換します。詳細については、「certpk12」(103 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 は、検証目的で certreq ツールと certauth ツールを提供しています。このツールを使用すると、ユーザは認証局として機能できるので、認証局署名済み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　証明書要求を認証局に送信します。

5　認証局署名済みの証明書が認証局から返信されたら、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格納します。

サードパーティ・ツールによる証明書の取得

多くのサードパーティ PKI ベンダといくつかのブラウザには、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あります。これらのユーティリティは、一般的にグラフィカル・ウィザードで、一連の質問に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済み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。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」(103 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、証明書の要求と認証を行うツールを提供しています。このツールは *%SYBASE%¥%SYBASE_OCS%¥bin* ディレクトリにあります。certreq では、パブリック・キーとプライベート・キーのペアと、証明書要求を生成します。certauth では、サーバ証明書要求を認証局署名済み証明書に変換します。

警告！ certauth は、テスト専用で使用します。Sybase では、商用認証局サービスを利用することをおすすめします。これは、ルート証明書統合の保護を提供し、また幅広く認められている認証局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認証に使用することが促進されるからです。

次の手順 1 ～ 5 に従って、サーバの信頼済み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 つの手順すべてを行い、テスト用の信頼済みルート証明書を作成します。テスト用の認証局証明書 (信頼済みルート証明書) を作成したら、手順 3 ～ 5 を繰り返してサーバ証明書に署名します。

❖ サーバの信頼済み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、証明書要求を認証局の自己署名証明書 (信頼済みルート証明書) 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、証明書要求を認証局署名済み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、ベンダが提供する暗号モジュールがある場合にのみ実行されます。この暗号モジュールでは、RSA と DSA アルゴリズムを使用して証明書要求を構築します。

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、『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認証局 (CA) 署名済み証明書に変換します。

構文

certauth [-r] [-C] [-Q] [-K] [-O] [-P] [-T]
[-r]
[-C *caCert_file*]
[-Q *request_filename*]
[-K *caKey_filename*]
[-O *SignedCert_filename*]
[-P *caPassword*]
[-T *valid_time*]
or certauth -v

-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 が指定されている場合は、認証局の証明書要求の名前を指定します。指定されていない場合は、認証局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*

認証局のプライベート・キーの名前を指定します。

-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* には自己署名ルート証明書が入ります。-r オプションを使用しない場合は、*SignedCert_filename* は *caCert_file* によって署名される証明書です。

-P *caPassword*

プライベート・キーの復号化に使用する認証局のパスワードを指定します。

-T *valid_time*

署名付き証明書の有効期間を指定します。有効期間は日単位です。

-v

certauth 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。

この例では、認証局の証明書要求 (*ca_req.txt*) を、プライベート・キー (*ca_pkey.txt*) を使用して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、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。信頼済みルート証明書を作成してから、これを使ってテスト環境の多くのサーバ証明書に署名します。

この例では、サーバ証明書要求 (*srv5_req.txt*) を証明書に変換して、有効期間を 180 日に設定します。この例では、認証局の証明書 (*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、署名済み証明書を *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
ByqGSM44BAMFAAMxADAuAhUAhM2IcnlpSavQtXFzXJUCoOmNLpkCFQDtE8RUGuo8

         ZdxnQtPu9uJDmoBiUQ==


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

使用法

- Adaptive Server が認識するサーバ証明書ファイルを作成するには、署名済み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、署名済み証明書ファイル *sybase_srv5.crt* の最後に *srv5_pkey.txt* を貼り付けます。
- サーバが起動時にロードできる信頼済みルート・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。

SSL ベースのセッションに必要なファイルを、SSL 対応 Adaptive Server の起動に使用します。

作成した認証局のルート証明書は、複数のサーバ証明書に署名するのに使用できます。

参照

certreq

certreq

サーバ証明書要求と対応するプライベート・キーを作成する。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*] または

certreq -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する必要があります。

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 共通名はサーバ名と同じにしてください。

input_file と呼ばれるサンプル・ファイルについては、101 ページの例 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ないので、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た後、さらに情報を入力するためのプロンプトが certreq ユーティリティから表示されます。

```
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*) とそのプライベート・キー (*server_pkey.txt*) を作成するものです。プライベート・キーは、*password* によって保護されます。

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> は大文字と小文字を区別し、上記と同じでなければなりません。
- “=” は必須です。有効な *value* は、文字または数字で開始し、単一のワードにしてください。*value* の中には、スペースを入れないでください。
- *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。これは certauth ツールで受け入れ可能な入力として使用されて、要求が認証局の署名済み証明書に変換されます。

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*]

certpk12 -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、-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 がオンの場合には、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 がオンの場合には、PKCS #12 ファイルからプライベート・キーをインポートしてからテキスト・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、パスワードを使用してエクスポートする PKCS #12 ファイルを暗号化します。-I がオンの場合は、パスワードを使用して、インポートする PKCS #12 ファイルを解読します。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* の解読に使用するパスワードです。*pk12password* は最終の *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 2000--
```

例 2

この例では、証明書とプライベート・キーのある 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* を使用して *caRSApkey_new.txt* を保護し、*pk12password* を要求して *caRSA.p12* ファイルを解読します。

```
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 2000--
```

注意 例 1 と例 2 を実行した後には、*caRSA.crt* と *caRSA_new.crt* は同じになりますが、*caRSApkey.txt* と *caRSApkey_new.txt* はランダムに暗号化されるので異なります。

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* を使用して *caRSAcert.p12* を暗号化します。

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

-- Sybase PKCS#12 Conversion Utility certpk12 Thu Nov 9
16:55:51 2000--
```

例 4

この例では、証明書のある 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* を要求して *caRSAcert.p12* ファイルを解読します。

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

-- Sybase PKCS#12 Conversion Utility certpk12 Thu Nov 9
16:55:51 2000--
```

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 暗号化方式により暗号化された PKCS #12 ファイルだけです。
- 証明書要求者のプライベート・キーを署名済み証明書ファイルの最後に付加します。
- ファイルに *servername*.crt と名前を付けます。*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置きます。

  SSL 対応 Adaptive Server の起動には、このファイルが必要です。

参照

certreq と certauth

索引

M

N

O

P

S

W

か

け

さ

し

せ

そ

て

と

ね

は

ひ

ふ

へ

ほ

ろ