IBM® DCE for AIX® and Solaris
バージョン 3.2

管理ガイド - コア・コンポーネント

IBM® DCE for AIX® and Solaris
バージョン 3.2

管理ガイド - コア・コンポーネント

ご注意!

本書をご使用になる前に、595ページの『付録E. 特記事項』にある一般的な情報を必ずお読みください。

本書は、*IBM DCE for AIX and Solaris* バージョン 3.2 に適用されます。また、改訂版等で特に断りのない限り、これ以降のすべてのリリースにも適用されます。

本書の記述および本書に関連するソフトウェアの一部は、以下の会社または団体に帰属しています。

プログラムの使用者は、適用されるすべての法律、規則、規制を遵守すること、およびプログラムの配布者に対しても遵守を要求することに同意するものとします。これらの法律、規則、規制は、(i) ライセンス・プログラムまたは文書を輸出または再輸出する場合に適用される技術データの輸出または再輸出に関するもの、 (ii) ライセンス・プログラム、文書、技術データについての政府機関の権限を制限するために要求される次のような内容の「権利制限表示」をライセンス・プログラム、文書、技術データに添付することに関するものです。「権利制限表示」の内容は、次のとおりです。「米国政府による使用、複写、開示は DFARS 52.227-7013(c)(1)(i)-(ii); FAR 52.227-19; および FAR 52.227-14, Alternate III, または他の連邦政府法規にある同内容の条項で定めるところにより、制限されます。」

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

`http://www.ibm.com/jp/manuals/main/mail.html`

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

`http://www.ibm.com/jp/manuals/` の「ご注文について」をご覧ください。 (URL は、変更になる場合があります)

原 典: IBM® Distributed Computing Environment Version 3.2 for AIX® and Solaris Administration Guide -- Core Componets

発 行: 日本アイ・ビー・エム株式会社

担 当: ナショナル・ランゲージ・サポート

第1刷 2001.3

この文書では、平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体*)は、(財) 日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、
平成角ゴシック体™W5、平成角ゴシック体™W7

目次

図

表

本書について

本書、*IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド ― コア・コンポーネント* には、IBM® 分散コンピューティング環境 (DCE) を管理するための概念と手順が解説してあります。本書の中では、基本的な DCE 用語を紹介しています。 DCE 資料用の用語集は、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE 入門* に載っています。 *IBM DCE for AIX and Solaris バージョン 3.2: DCE 入門* は、 DCE 技術のハイレベルな解説を行うとともに、 DCE をサポートする資料について説明しています。

本書の対象読者

この手引きは、AIX® または Solaris 環境を以前に管理したことがあるシステム管理者およびネットワーク管理者を対象に書かれています。

適用範囲

この版は、IBM DCE 3.2 オファリングおよびその更新版に適用されます。詳細については、ご使用のソフトウェア・ライセンスを参照してください。

本書の目的

この手引きは、システム管理者およびネットワーク管理者が DCE の計画、構成、および管理を行うのを支援することを目的としています。この手引きを読むと、DCE を計画するためにシステム管理者は何をするべきかが分かります。ご使用のシステムに DCE ソース・コードを構築後は、実行可能ファイルのインストールおよび DCE の構成にこの手引きをご利用ください。ご使用のプラットフォームでの DCE ソース・コードのインストールや構築については、 *IBM DCE for AIX バージョン 3.2: 概説とインストール AIX 版* または *IBM DCE for Solaris バージョン 3.2: 概説とインストール Solaris 版* を参照してください。

本書の使用方法

IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド は、2 冊で構成され、それぞれが以下のような部に分かれています。

- *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド ― 入門*
  - 第 1 部 DCE 管理の紹介
  - 第 2 部 DCE の構成と起動
- *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド ― コア・コンポーネント*
  - 1ページの『第1部 DCE 制御プログラム』
  - 65ページの『第2部 DCE 管理タスク』
  - 101ページの『第3部 DCE ホストとアプリケーションの管理』
  - 153ページの『第4部 セル・ディレクトリー・サービス』
  - 261ページの『第5部 DCE 分散タイム・サービス』

– 315ページの『第6部 DCE セキュリティー・サービス』

関連資料

分散コンピューティング環境の詳細については、以下の資料を参照してください。

- *IBM DCE for AIX and Solaris バージョン 3.2: DCE 入門*
- *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス*
- *IBM DCE Version 3.2 for AIX and Solaris: Application Development Reference*
- *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Introduction and Style Guide*
- *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components*
- *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合*
- *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Directory Services*
- *OSF DCE/File-Access Administration Guide and Reference*
- *OSF DCE/File-Access User's Guide*
- *IBM DCE Version 3.2 for AIX and Solaris: Problem Determination Guide*
- *OSF DCE Testing Guide*
- *OSF DCE/File-Access FVT User's Guide*
- *Application Environment Specification/Distributed Computing*
- *IBM DCE for AIX バージョン 3.2: リリース情報 AIX 版*
- *IBM DCE for Solaris バージョン 3.2: リリース情報 Solaris 版*

DCE 3.2 for AIX and Solaris のマニュアルについて詳しくは、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE 入門* を参照してください。

本書で使用される規則

このガイドでは印刷字体に関して次のような規則を使用します。

太字 **太字**の語または文字は、コマンド、オプション、およびパス名のように、表示されているとおりに使用する必要があるシステム要素を示します。

イタリック
イタリック の語または文字は、ユーザーが指定する必要がある変数値を示します。*イタリック* は、新しい DCE 用語を紹介するときにも使用されます。

モノスペース
システムによって表示される例および情報は、**モノスペース**で表示されます。

[ ] 大括弧は、フォーマットおよび構文の記述内のオプション項目を囲みます。

{ } 中括弧は、フォーマットおよび構文の記述内の 1 つの項目をそこから選択する必要があるリストを囲みます。

| 垂直線は、選択項目のリスト内の項目を分離します。

< > 不等号括弧は、キーボード上のキーの名前を囲みます。

... 水平の省略符号の点線は、先行する項目を 1 回または複数回反復できることを示します。

この手引きではキー操作に関して次のような規則を使用します。

<Ctrl-*x*> または ^*x*

キーの名前が後ろに続いている、<Ctrl-*x*> または、 ^*x* という表記は、制御文字シーケンスを示します。たとえば、<Ctrl-C> は、制御キーを押したまま <C> を押すことを意味します。

<Return>

<Return> という表記は、端末装置またはワークステーション上にある、Return または Enter という語あるいは左矢印のラベルが付いたキーを示します。

問題の報告

ソフトウェアまたは資料に問題が生じた場合は、ソフトウェア・ベンダーのカスタマー・サービス部門にご連絡ください。

DCE の資料にあるディレクトリーおよびファイルのパス名

この手引きで参照するディレクトリーおよびファイルのパス名のリストについては、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド － 入門* と *OSF DCE Testing Guide* を参照してください。

第1部 DCE 制御プログラム

第1章 DCE 制御プログラムの紹介

DCE は、異種ネットワーク・コンピューター間で分散アプリケーションの開発と実行をサポートする 1 組の統合されたサービスです。それぞれの DCE 環境 (セルと呼ばれます) は、少なくとも次のコア DCE サービスを保守します。

- DCE スレッド
- DCE ホスト・サービス
- DCE セル・ディレクトリー・サービス
- DCE タイム・サービス
- DCE セキュリティー・サービス

DCE スレッドを除くすべてのコア・サービスには、さまざまな方法での管理が必要です。 DCE タイム・サービスなどのサービスは、一度設定した後は実質的に介入が必要ないので、 CDS および DCE セキュリティーのようなサービスより通常は少ない管理で済みます。 DCE セルがわずかなコンピューターとそのユーザーで構成されている場合には、必要な管理タスクを実行するために、各ホストにログオンして、ユーザー、プログラム、およびホスト・システムの命名、時刻、およびセキュリティーに関する要件を管理することができます。しかし、大部分のセルは、数百、数千ものコンピューターとそのユーザーから構成されています。

したがって、このようなセル内のコア・サービスは広範囲にわたり、そのうちのいくつかのサービスは複数の異種システム間で複製されたり、分割されたりするので複雑になります。 DCE ホスト・サービスなどのいくつかのサービスは、セル内のすべてのコンピューターに存在します。このような大規模なオペレーションでは、DCE 管理機能がどこにあるかに関係なく、セル内のすべてのポイントからこの機能に一貫性のある均質のアクセスを提供する管理インターフェースが必要になります。つまり、管理操作は、実行されるプラットフォームに関係なく、常に一貫して、しかも予想通りに機能する必要があります。

DCE 制御プログラム (**dcecp**) は、 DCE セル内の任意の点からほとんどすべての DCE 管理機能に対して、常に一貫しており、ポータブルで、拡張可能な完全なアクセスを提供しており、これらの要件を満たしています。 **dcecp** プログラムは、従来はさまざまなコンポーネント制御プログラムを使用することによって実行されてきた、ほとんどのオペレーションを実行します。

dcecp プログラムは、複雑な DCE オペレーションを実行するための多くのタスク・オブジェクト を提供することによって、管理をさらに能率化します。たとえば、セルへのホストの追加には、レジストリーへのホスト・プリンシパルの追加、さまざまなセキュリティーのグループおよび組織へのプリンシパルの追加、アカウントの作成、CDS へのホスト情報の追加、および CDS ディレクトリーでのいくつかの ACL の設定などが必要となります。これらのオペレーションはすべて、単一のタスク・オブジェクトを使用して実行することができます。

注: dcecp は各種の言語環境プログラムをサポートするように設計されています。それは従来の制御プログラム (cdscp、 dtscp、 rpccp、 acl_edit、 rgy_edit、 sec_admin) に取って代わるものです。これらの従来のプログラムは国際的に使用できるようには設計されていなかったので、英語以外の環境で使用すると予期しないまたは望ましくない結果になることがあります。

dcecp は英語以外のデータをサポートしますが、いくつかの制約事項があります。 **dcecp** の、ストリング範囲などストリングを扱うコマンドは、文字ベースではなくバイト・ベースのセマンティクスになっています。これらのコマンドを DCE ポータブル文字セット以外の文字に使用すると、望ましくない結果になる可能性があります。

柔軟性があり、ポータブルで拡張可能な管理

dcecp 制御プログラムは、ツール・コマンド言語を意味する Tcl (「ティクル」と発音します) と呼ばれるポータブルなコマンド言語で構築されます。この言語は、カリフォルニアのバークレーにあるカリフォルニア大学の John K. Ousterhout によって開発されました。大部分のコンピューターには、何らかのコマンド言語が備えられているため、ユーザーは柔軟性があり拡張可能な方法でシステム機能にアクセスし、それを使用することができます。たとえば、多くの UNIX® スタイルのシステムはシェル言語インタープリターを備え、 Digital Equipment Corporation の OpenVMS オペレーティング・システムはディジタル・コマンド言語 (DCL) を備えています。ただし、こうしたコマンド言語は必ずしもポータブルではありません。あるコマンド言語に基づくコマンドおよびスクリプトが他のコマンド言語環境では動作しない場合があります。

一方、Tcl は、プラットフォームから独立したコマンド言語であるため、 DCE がインストールされているすべてのシステムで作動します。 Tcl コマンド・インタープリターおよびそれを使用する DCE 制御プログラムは、 DCE ソフトウェアの一部として提供されます。

DCE 制御プログラムと DCE 制御プログラム言語の両方により、 DCE 管理者は次の重要な利点が得られます。

- 日常的な DCE オペレーションは、単一の管理インターフェース内からすべて実行することができます。
- 大部分の DCE 管理オペレーションは、任意の DCE プラットフォームから一貫して実行されるので、管理者は、セル内の任意の DCE システムからほとんどすべての DCE オペレーションを管理することができます。 UNIX システムではない DCE プラットフォームでは、すべての DCE 制御プログラム・ファイル・オペレーションを処理できない場合があります。
- **dcecp** プログラムには、**clearinghouse**、**principal**、および **endpoint** などの名前を持つ管理オブジェクト が用意されています。このような直接的な方法によって、DCE 管理は直観的であり、一貫性を保っています。現在のところ、これはオブジェクト指向のように見えるだけですが、真のオブジェクト指向の管理インターフェースへの重要なステップです。
- タスク・オブジェクト (複雑な DCE オペレーションを実行する高水準の **dcecp** スクリプト) によって、DCE 管理者のトレーニング要件が低減されます。日常的な DCE 管理用タスクを実行するために DCE 専門家である必要はありません。
- **dcecp** オペレーションを、Tcl 内で提供される、より一般的なコマンドと共に使用すれば、提供されたタスク・オブジェクトを新しい使用法に応用したり、新しいタスク・オブジェクトまたはスクリプトを作成したりすることができます。
- **dcecp** 言語では、オペレーション能力を高めるための、変数、**if** ステートメント、ループ機能およびその他のプログラミング・オペレーションを使用すること

ができます。たとえば、ループ機能によって、ユーザー、サーバー、または CDS 項目などの複数のオブジェクトに対してオペレーションを反復することができます。

- スクリプトは変更せずに外部プラットフォームに移動できるので、管理者はツールを容易に共用することができます。たとえば、複数のセルがあるエンタープライズで **dcecp** スクリプトを使用すれば、エンタープライズ全体に共通のセル構成を伝搬することができます。

DCE 制御プログラムは、DCE コア・コンポーネントの大部分の局面を管理するために使用できる、管理インターフェースです。 **dcecp** を使用しても、DCE のすべての局面を管理できるわけではありません。たとえば、**dcecp** は DFS™ を制御できません。

1ページの『第1部 DCE 制御プログラム』の各章は、**dcecp** を使用して、 DCE 環境内のコア・サービスを管理する方法について説明しています。さらに、コマンド行で Tcl 構成を使用したり、スクリプトとして独自にカスタマイズした操作を作成したりすることによって、さまざまなオペレーションを行う方法についても説明しています。 X11 ウィンドウ・システム用の Tcl またはそれに付随するツールキット (*Tk* と呼ばれます) については詳細な説明は行いません。

DCE 管理オブジェクト

DCE セルは、管理が必要な多くのものから構成されます。たとえば、 CDS サーバー (クリアリングハウス)、 DTS クロック、およびサーバー・ロケーション情報はすべて、何らかの方法で管理が必要な、DCE セル内のエンティティーです。 DCE 制御プログラムは、 DCE の管理エンティティーすべてを個別の管理オブジェクトとして扱います。

エンティティーでのオペレーションは、そのオブジェクト 名をある種のオペレーションとともに呼ぶことによって行います。したがって、たとえば、 DTS クロックの時刻を検査するには、次の例に示すように、オブジェクトの名前 (**clock**) と必要なオペレーション (**show**) を呼び出します。

```
dcecp> clock show
1994-09-23-10:46:42.016-04:00I-----
dcecp>
```

DCE 内の各管理エンティティーには、DCE 制御プログラム内に、対応する管理オブジェクトがあります。例を挙げると、**clearinghouse** オブジェクトを使用することによって、セル内の CDS クリアリングハウス・オペレーションを管理できます。また、DCE ホスト上のアプリケーション・サーバーおよびその構成情報は、**server** オブジェクトを使用して管理します。時刻情報は、**utc** オブジェクトを使用して、比較し、操作します。 DCE セル内のユーザーは、**user** オブジェクトを使用して管理します。これらは、**dcecp** 管理オブジェクトのほんの数例です。すべてのオブジェクトのリストは、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* にあります。

DCE 制御プログラムの使用法

この項では、DCE 制御プログラムを開始し停止する方法、およびオペレーションを実行する方法について簡単に説明します。これらのトピックに関する補足情報は、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* に記載されています。

サーバーの開始および停止

dcecp オペレーションは、オペレーティング・システム・プロンプトから、または DCE 制御プログラム内から直接に入力することができます。 1 つまたは 2 つの単純な **dcecp** オペレーションを実行するだけならば、オペレーティング・システム・プロンプトから直接それらを呼び出すことができます。

いくつかのオペレーションを実行する場合には、DCE 制御プログラムを起動した後、 **dcecp** プロンプトからオペレーションを入力することができます。この方法にはいくつかの利点があります。

- **dcecp** の初期設定が個別のオペレーションごとではなく一度だけ行われるので、複数のオペレーションの場合はより効率的です。
- 制御プログラムがオペレーションをヒストリー機能に格納するので、再呼び出しと再利用が可能です。
- **dcecp** コマンドによる各コマンドの前に、余分のキー入力を行う必要がありません。

次の例は、DCE 制御プログラムを起動して、 **directory** オペレーションを行う方法を示しています。

```
% dcecp
dcecp> directory create /.:/hosts/appserver2
dcecp>
```

DCE 制御プログラムの使用が完了したら、 **exit** オペレーションまたは **quit** オペレーションを使用してプログラムを停止し、オペレーティング・システム・プロンプトに戻ってください。次の例は、**exit** オペレーションの使用法を示しています。

```
dcecp> exit
%
```

dcecp オペレーションの呼び出し

単一の **dcecp** オペレーションを実行する場合は、オペレーティング・システム・プロンプトから直接呼び出すことができます。次に示すように、 **dcecp** コマンドと **-c** (コマンド行オペレーション) フラグの後に、必要なオペレーションを入力します。

```
% dcecp -c directory list /.:/subsys -simplename
HP applications dce sales eng admin accts
% dcecp -c cell show
{secservers
 /.../my_cell.goodco.com/subsys/dce/sec/master}
{cdsservers
 /.../my_cell.goodco.com/hosts/krypton}
{dtsservers
 /.../my_cell.goodco.com/hosts/mars}
{hosts
 /.../my_cell.goodco.com/hosts/earth
```

```
 .../my_cell.goodco.com/hosts/jupiter
 /.../my_cell.goodco.com/hosts/kyrpton
 /.../my_cell.goodco.com/hosts/mars
 /.../my_cell.goodco.com/hosts/mercury
 /.../my_cell.goodco.com/hosts/neptune
 /.../my_cell.goodco.com/hosts/pluto
 /.../my_cell.goodco.com/hosts/saturn
 /.../my_cell.goodco.com/hosts/uranus
 /.../my_cell.goodco.com/hosts/venus}
%
```

また、; (セミコロン) をコマンド区切り記号として使用し、 "" (二重引用符) でオペレーションを囲むことによって、いくつかの限定された複数のオペレーションを入力することもできます。次の例は、プリンシパルをレジストリーに追加した後、プリンシパルが正しく追加されたかどうかを検査しています。

```
% dcecp -c "principal
create S_Preska ; principal show S_Preska"
{fullname {}}
{uid 28}
{uuid 0000001c-dc77-21cd-b700-0000c08adf56}
{alias no}
{quota unlimited}
%
```

-c オプションを指定して **dcecp** コマンドから複数のオペレーションを入力する場合は、オペレーションの結果が、シェルではなく **dcecp** インタープリターに戻るので注意が必要です。次の場合は、最後のオペレーション (group list users) の結果だけがシェルに戻ります。

```
% dcecp -c "group list staff; group list managers; group list users"
/.../ward_cell.osf.org/P_Pestana
/.../ward_cell.osf.org/R_Parsons
/.../ward_cell.osf.org/L_Jones
/.../ward_cell.osf.org/S_Preska
/.../ward_cell.osf.org/N_Long
/.../ward_cell.osf.org/D_Witt
/.../ward_cell.osf.org/C_Pilat
 .
 .
 .
%
```

この問題は、次のようにすることによって克服することができます。

```
% dcecp -c "puts [group list staff]; puts [group list manager];
puts [group list users]"    .    .    . %
```

dcecp スクリプトを起動するためには、**-c** 引き数は省略し、スクリプトの名前は含めてください。次の例は、セル内のすべてのホストの名前をアルファベット順にリストするスクリプトを起動します。

```
% dcecp list_hosts
earth
jupiter
krypton
mars
mercury
neptune
planets
pluto
```

```
saturn
uranus
venus
%
```

複雑なオペレーションまたは複数のオペレーションを呼び出す必要があるときは、**dcecp** 内からオペレーションを呼び出すことができます。このプログラムには、前のオペレーションの再呼び出しおよび再利用に便利なヒストリー機能およびコマンド行編集機能があります。次の例は、 **dcecp** を起動して、DCE セルに新しいユーザーを追加します。

```
% dcecp
dcecp> principal create J_Jones
dcecp> group add users -member J_Jones
dcecp> organization add staff -member J_Jones
dcecp> account create J_Jones -group users -organization staff ¥
> -password change.me -mypwd mxyzptlk
dcecp>
```

すべての **dcecp** オブジェクト、オペレーションおよびオプションの名前は、対話モードで使用する場合、最短の固有のストリングに省略することができます。これらの名前は、省略形を考慮して選択されているので、固有の省略形は通常、1 文字または 2 文字になります。

スクリプト内でのオブジェクトまたはコマンドの省略形の使用は、スクリプトの移植性を制限するので、避けてください。ユーザー自身のコマンドを定義した場合、省略形の固有性が変化する可能性があり、コマンド名またはオブジェクト名が不明確になります。

dcecp による他の処理

DCE 制御プログラムは、単純なコマンドから複雑なコマンドまでの、広範囲のコマンドを受け付けます。複雑なコマンドによって、より高い機能と多様性が提供されます。単純なコマンドは作成が容易ですが、通常は単一のオブジェクトに対し 1 つのオペレーションの実行に限定されます。したがって、単純コマンドを入力することは常に可能ですが、オペレーションをいくつかの、または多くのオブジェクトで反復したい場合、あるいは特定の条件のもとでのみ何らかのオペレーションを実行したい場合があります。たとえば、ある項目が CDS ディレクトリーにすでに存在する場合にのみ、別のある項目を CDS に追加したいことがあります。**dcecp** プログラムでは、多くのプログラミング言語およびシェル言語に通常備わっている要素をまねた、 Tcl の組み込みコマンドを利用することによってこれを行うことができます。

DCE 制御プログラムには、コマンドの実行を制御する、 C に似た多くの構成が含まれています。例を挙げると、条件付き実行のための **if** ステートメント、さまざまな条件のもとでオペレーションを反復するために使用する **while**、 **for** および **foreach** などのループ・コマンド、さまざまなパターンに対して値をテストする **case** コマンド、および独自のカスタマイズ済みコマンドを作成するための **proc** コマンドなどがあります。

また、DCE 制御プログラムには、″″ (引用符)、{ } (中括弧)、 [ ] (大括弧)、および ¥ (円記号) といった、他の構文要素も含まれており、要素をグループにまとめたり、特殊文字の解釈を制御したりするために使用されます。

多くの機能はスクリプト内で使用するために設計されていますが、対話モード・オペレーションでいくつかの構成と要素 (特に、引用符、中括弧、大括弧、および円記号) を使用していることに気付くはずです。オペレーションを対話モードで実行するか、またはスクリプトを使用して実行するかを選択する必要があります。一般的に、複雑さと再利用の可能性が、判断の基準になります。

基本的な DCE 制御プログラムと Tcl を示す 2 つの簡単な例を見ることにします。 **dcecp** オペレーションは、次のように非常に単純な場合があります。

```
dcecp> account modify N_Long -expdate
1996-06-30
dcecp>
```

このオペレーションは、DCE セキュリティー・サービス・レジストリー内の情報を変更します。ここでは、コマンド行で指定したプリンシパル (**N_Long**) のアカウント有効期限日を変更します。このオペレーションを 1 つか 2 つのプリンシパルに関して実行する場合は比較的単純ですが、多くのプリンシパルのアカウント有効期限日を変更する場合のオペレーションは、複雑になります。

組織が 6 人の臨時作業員を雇っていて、関連するプロジェクトが 3 カ月延長されたとします。 **account modify** オペレーションを 6 回実行するのではなく、**dcecp foreach** コマンドを使用すれば、リストの項目ごとにアクションをループ (反復) することができます。

```
dcecp> foreach i {N_Long L_Jones P_Sawyer ¥
> D_Witt M_Dougherty S_Preska} { ¥
> account modify $i -expdate 1996-06-30 }
dcecp>
```

この例では、**foreach** ループ・コマンドには、変数、リスト、および本体という 3 つの引き数があります。変数 **i** は、リスト (**N_Long**、 **L_Jones** など) 内の各項目を順番に置き換えます。 **foreach** コマンドは、リスト内の項目ごとに、本体 (**account modify $i -expdate 1996-06-30**) を実行します。本体の中の **$i** 変数は、リスト内のすべての項目が使用されるまで、リスト内の各プリンシパル名の値をとります。ループ・コマンドに関する詳細については、 36ページの『ループによるスクリプトの実行の制御』を参照してください。

この例は、その他の重要な構文規則をいくつか示しています。 DCE 制御プログラムは、スクリプト本体などのコマンド引き数の開始と終了の位置を判別するために、 { } (中括弧) を使用します。たとえば、**foreach** コマンドには、変数名、リスト、およびスクリプト本体という 3 つの引き数があります。通常、コマンド引き数はスペースで区切ります。 **dcecp** がリスト要素間のスペースを誤って引き数の区切り記号として解釈しないように、リストを中括弧で囲んで、スペースの特殊な解釈を使用不能にします。したがって、すべてのリスト要素が 1 つの引き数として解釈されます。同様に、スクリプト本体内の各要素を囲むためにも中括弧を使用します。

中括弧は **dcecp** がコマンドの完了を判別できるようにするためにも使用されます。コマンドが完了していない場合、左中括弧が右中括弧より多くなります。最初の行の末尾に右中括弧がないと、コマンド入力が続くことを **dcecp** に知らせるので、 **dcecp** は 2 番目のプロンプト (>) を表示して入力を指示します。同様に、2 行目の最後の左中括弧は、コマンドの入力が終了していないことを示します。これによって、¥ (円記号) 行折り返し文字を使用しないで、行を折り返すことができま

す。 DCE 制御プログラムは、 3 行目の最後の右中括弧の後で **<Return>** が押されると、そのコマンドを実行します。中括弧の詳細については、21ページの『第2章 DCE 制御プログラム・コマンド言語の使用』に説明があります。

ここで、組織に臨時作業員 が 6 人ではなく 50 人いて、 (すべての人が **temps** と呼ばれる 1 つのグループに属しています) 3 か月のアカウント拡張を追加する必要があるとします。この場合も **foreach** コマンドを使用しますが、リスト内に 50 個のプリンシパルすべてを直接書き込む代わりに、 **dcecp group list temps** オペレーションを使用して、リストを生成させます。

```
dcecp> foreach i [group list temps] {
> account modify $i -expdate 1996-06-30 }
dcecp>
```

この例で、group list temps オペレーションは [ ] (大括弧) で囲まれています。この技法は、*コマンド置換* と呼ばれ、大括弧内のコマンドを、そのコマンドから戻された結果で置き換えます。 **group list temps** オペレーションの結果、次に示すような有効な Tcl リストが生成されます。

```
dcecp> group list temps
N_Long
L_Jones
P_Sawyer
D_Witt
M_Dougherty
S_Preska
 .
 .
 .
J_Jones
```

ここでは、**dcecp** の実際的な用法をハイレベルな観点から見てきました。もちろん、見てこなかったものもたくさんあります。次の章では、 DCE 管理のために使用できる **dcecp** オペレーションのいくつかをより細かく見ていきます。 **dcecp** は Tcl に基づいていますが、Tcl には、本書では説明しない他のコマンドや変形コマンドがあることに注意してください。したがって、すべてのコマンドの詳細については、標準の Tcl 資料を参照してください。

対話式コマンドまたはスクリプトのいずれを使用するか

コマンドを対話式に入力すべきか、またはスクリプトを使用すべきかを決定する絶対的な基準はありません。しかし、一般的に、1 つまたは 2 つのタスクを実行するような単純なオペレーションの場合には、対話式の入力を選択すべきです。対話式オペレーションの典型的な例を次に示します。

```
dcecp> directory create /.:/printers

dcecp> account show w_shakespeare

dcecp> server start /.:/hosts/curley/config/srvrconf/BBSserver
```

次の例は、多少複雑なので、最初はスクリプトとしての実行を選択することもできます。

```
foreach i [group list temps] {
   account modify $i -expdate 1996-06-30}
```

頻繁に使用するオペレーションをスクリプトとして (ファイルに) 保管することによって、反復するタスクまたは複雑なタスクを自動化したり、変更できるように手近に保持したり、後から別の状態で使用したりすることができます。いずれの方法を選択しても、 **dcecp** と Tcl に慣れると、かなり複雑なオペレーションも対話式で入力できるようになるはずです。スクリプトの作成と実行の方法の詳細については、18ページの『dcecp セッションのカスタマイズ』を参照してください。

コマンド行の編集

ここまでは、対話式 **dcecp** コマンドを入力する基本的な方法をいくつか見てきました。しかし、入力中のコマンドを編集したり、あるいは以前に入力したコマンドを再呼び出しして変更したりする必要がある場合があります。 DCE 制御プログラムには、コマンドを編集するためのいくつかの方法が用意されています。現行のコマンド行を編集するには、コマンド行編集機能を使用することができます。あるいは、**history** コマンドを使用して、以前に使用したコマンドを再呼び出しして、編集し、再発行することができます。

現行コマンド行の編集

ksh または `emacs` 編集コマンドに似た、制御文字およびエスケープ・シーケンスを入力することによって、コマンド行を `dcecp` に送信する前に編集することができます。 **<Ctrl-*x*>** (この *x* は文字) などの*制御文字* は、**<Ctrl>** (または **<Control>**) を押したまま文字キーを押します。たとえば、**<Ctrl-A>** は、 **<Ctrl>** と **<A>** を同時に押します。エスケープ・シーケンス の入力には、 **Escape** の後に 1 つまたは複数の文字を押します。エスケープ・シーケンスでは、 **<Escape>** は **ESC** とも表されます。たとえば、**<ESC f>** となります。エスケープ・シーケンスでは、大文字小文字の区別がされます (制御文字では、大文字と小文字が区別されません)。したがって、**<ESC F>** と **<ESC f>** とは異なります。

編集コマンドは、行の先頭だけではなく、行の任意の位置に入力することができます。さらに、リターン・キーも行末だけでなく、行の任意の位置で使用することができます。

大部分の編集コマンドは、反復カウント*n* を受け入れます。*n* は数字です。反復カウントを指定するには、**<Escape>**、数字、次に実行したいコマンドを入力します。たとえば、**<ESC 4><Ctrl-f>** は、4 文字分、下方に移動させます。以下の説明では、反復カウントを受け入れるコマンドには、 [*n*] を付けて示してあります。

次の制御文字が受け入れられます。

<Ctrl-A>
行の先頭に移動する

<Ctrl-B>
左 (後ろ方向) に移動する [*n*]

<Ctrl-D>
文字を削除する [*n*]

<Ctrl-E>
行の末尾に移動する

<Ctrl-F>
右 (前方向) に移動する [*n*]

<Ctrl-G>
ベルを鳴らす

<Ctrl-H>
カーソルより前の文字を削除する (**<Backspace>**) [*n*]

<Ctrl-I>
ファイル名を完了する (**<Tab>**)。下の説明を参照。

<Ctrl-J>
行を処理する (**<Return>**)

<Ctrl-K>
行 (または列 [*n*]) の終わりまで kill (削除) する

<Ctrl-L>
行を再表示する

<Ctrl-M>
行を処理する (代替 **<Return>**)

<Ctrl-N>
ヒストリーから次行を入手する [*n*]

<Ctrl-P>
ヒストリーから前行を入手する [*n*]

<Ctrl-R>
ヒストリーを上方探索 ([*n*] の場合は下方探索) してテキストを探す。テキストが上矢印で始まる場合は行を開始しなければならない。

<Ctrl-T>
文字を転置する

<Ctrl-V>
編集コマンドであっても、次の文字を挿入する

<Ctrl-W>
マークまでワイプ (除去) する

<Ctrl-X><Ctrl-X>
現在位置とマークを交換する

<Ctrl-Y>
最後に kill (削除) されたテキストを復元する

<Ctrl-[>
エスケープ・シーケンス (**<Escape>**) を開始する

<Ctrl-]>*c*
次の文字*c* まで前進する

<Ctrl-?>
カーソルより前の文字を削除する (**<Delete>**) [*n*]

以下のエスケープ・シーケンスが受け入れられます。

<ESC><Ctrl-H>
前にある文字を削除する (**<Backspace>**) [*n*]

<ESC DEL>
前の語を削除する (**<Delete>**) [*n*]

<ESC SPC>
マークを設定する (**<Spacebar>**)。**<Ctrl-X><Ctrl-X>** と **<Ctrl-Y>** を参照

<ESC .>
前の行から最後 (または [*n* 番目]) の語を入手する

<ESC ?>
可能な完了を示す。以下の説明を参照。

<ESC <>
ヒストリーの先頭に移動する

<ESC >>
ヒストリーの終わりに移動する

<ESC b>
1 つ前方の語に移動する [*n*]

<ESC d>
カーソルの下の語を削除する [*n*]

<ESC f>
1 つ後方の語に移動する [*n*]

<ESC l>
語を小文字にする [*n*]

<ESC u>
語を大文字にする [*n*]

<ESC y>
最後に kill (削除) されたテキストを復元する

<ESC w>
マークまでのエリアを復元可能にする

<ESC *nn***>**
反復カウントを数値*nn* に設定する

場合によっては、既存の端末キーの割り当てがこれらの **dcecp** 制御キーより優先することがあります。特に、使用環境における **erase**、 **kill**、**eof**、**intr**、**quit** および **susp** に対する割り当ては、同じ制御キーに対する **dcecp** の割り当てに優先します。ほとんどの場合、制御キーは端末では解釈されませんが、 **dcecp** には渡されます。ただし、例外の 1 つとして **lnext** (literal next) があります。これは、次に入力される文字を引用符で囲みます。 **lnext** に割り当てられている制御キーをタイプすると、端末がそれを解釈し、次に入力された文字を **dcecp** に渡します。

DCE 制御プログラムは、ファイル名完了も提供します。ルート・ディレクトリーに以下のファイルが入っているとします。

```
bin vmunix
core vmunix.old
```

rm /v とタイプして **<Tab>** を押すと、コマンド・プロセッサーは、**munix** を追加することによって、名前をできる限り完成させます。この例の名前は固有ではないため、ビープ音が鳴ります。そこで、**<Escape>**、続いて **?** (疑問符) を入力すると、 2 つの選択肢が表示されます。以下のようにピリオド (これは名前を固有なものにします) の後に続けて **<Tab>** を入力すると、コマンド・プロセッサーがファイル名を完成させます。

```
rm /v <Tab>munix.<Tab>old
```

この例では、固定幅字体のフォントは、コマンド・プロセッサーによって自動的に入力されるテキストを示します。

history コマンドによるコマンド行の編集

対話式コマンドを入力しているとき、以前に入力したコマンドを再呼び出しして使用したい場合があります。 CDS ディレクトリー内のオブジェクトをリストした後、そのうちの 1 つを変更するとします。オブジェクトをもう一度リストして、変更が有効になったことを確認する必要があります。あるいは、**history** コマンドを使用して、以前に使用したコマンドを再呼び出しして、編集し、再発行することができます。ヒストリー機能は、対話式コマンドのみを保管します。スクリプトから発行されたコマンドは保管されないため、再呼び出しすることはできません。

history コマンドは、何を実行したいかによって、さまざまな引き数を使用します。引き数を指定しないで **history** を入力すると、現在の **dcecp** 呼び出し中に入力したヒストリー・リスト上のすべてのコマンド (*イベント* と呼ばれます) が表示されます。

```
dcecp> history
     1  principal create wardr -fullname {Ward Rosenberry} ¥
          -quota unlimited
     2  group add users -member wardr
     3  organization add consultants -member wardr
     4  account create wardr -mypwd mxyptlk -password qwerty ¥
          -group users -organization consultants
           5 history
dcecp>
```

各ヒストリー・イベントは、前のイベントから独立しています。これは、再呼び出ししたコマンドが変数を使用している場合、その現行値は、最初にそのコマンドを入力したときと同じではない可能性があることを意味します。 **history** コマンド自体もヒストリー・イベントを生成します。

デフォルトでもって、ヒストリー・リストには 20 個の最新コマンドが保持されます。 **history keep** コマンドを使用すれば、ヒストリー・リストを長くしたり短くしたりすることができます。たとえば、次のコマンドは最新の 50 イベントを保持できるようにヒストリー・リストを長くします。

```
dcecp> history keep 50
dcecp>
```

イベントは、さまざまな方法で指定することができます。正数は、リスト内の最も古いイベントに相対したイベントを指定します。負数は、最新のコマンドに相対したイベントを指定します。以前のイベントの全部または一部に一致する文字を入力することによって、イベントを指定することもできます。

ヒストリー機能によって、以前のイベントを多くの方法で再利用することが可能となります。次に挙げるのは、使用できるヒストリー・コマンドのほんの数例にすぎません。

- **history redo** コマンドを使用することにより、以前のコマンドを、変更を加えずに、実行することができます。

```
dcecp> history
           1 directory show /.:/printers
           2 object create /.:/printers/ascii_printer1
           3 object create /.:/printers/ascii_printer2
           4 object create /.:/printers/ascii_printer3
           5 history
dcecp> history redo directory
directory show /.:/printers
 .
 . [output omitted]
 .
dcecp>
```

history コマンド内の各語の最初の固有の文字だけを入力することによって、入力の手間の大部分を省くことができます。たとえば、上記の例の **history redo directory** コマンドを次のように入力することができます。

```
dcecp> hi r d
directory show /.:/printers
 .
 . [output omitted]
 .
dcecp>
```

コマンドを再実行する他の方法としては、最新のコマンドを再呼び出しする **!!** と、特定のイベントを再呼び出しする **!***event number* とがあります。

- **history substitute** コマンドを使用すれば、以前のコマンドを修正して再実行することができます。このコマンドの一般的な使用法は、入力ミスの訂正です。このコマンドの構文は次のとおりです。

history substitute *old new* [*event number*]

event_number を省略すると、最新のコマンドが再実行されます。このコマンドは、再呼び出ししたコマンドの *old* 部分を *new* 情報で置き換えてから、実行します。

```
dcecp> history
           1 directory show /.:/printers
           2 object create /.:/printers/ascii_printer1
           3 object create /.:/printers/ascii_printer2
           4 object create /.:/printers/ascii_printer3
           5 directory show /.:/printers
           6 history
dcecp> hi s printer3 printer4 -3
object create /.:/printers/ascii_printer4
dcecp>
```

次に示すように、UNIX **csh** シェルのユーザーにおなじみの ^*old*^*new* 構文を使用すれば、最新のコマンドを再呼び出しして修正することもできます。

```
dcecp> ^4^5
object create /.:/printers/ascii_printer5
dcecp>
```

dcecp ヘルプ機能の使用法

DCE 制御プログラムは、いくつかの方法でヘルプ機能を提供します。

- DCE 制御プログラムが提供するオブジェクトのリストを表示したい場合には、次の例のように、**dcecp** プロンプトで help と入力してください。

```
dcecp> help
The general format of all dcecp commands is as follows:
  dcecp> <object> <operation> [argument] [options]

In addition to all of the standard tcl commands, dcecp supports many
commands to administer DCE objects.  A dcecp object or task represents
a DCE entity.  Type 'man dcecp_<command>' for more information.
All
of the following dcecp objects and tasks require an operation:
  account        cdsalias       dts            log            rpcprofile
  acl            cdscache       endpoint       name           secval
  attrlist       cdsclient      group          object         server
  aud            cell           host           organization   user
  audevents      cellalias      hostdata       principal      utc
  audfilter      clearinghouse  hostvar        registry       uuid
  audtrail       clock          keytab         rpcentry       xattrschema
  cds            directory      link           rpcgroup

Miscellaneous commands perform specific functions.  Type 'man dcecp'
for more information.  These commands take no operation:
  echo     errtext  login    logout   quit     resolve  shell

To list all dcecp objects:                    dcecp> help -verbose
To list all operations an object supports:    dcecp> <object> help
To list all options for an object operation:  dcecp> <object> help <operation>
For verbose information on a dcecp object:    dcecp> <object> help -verbose
For the manual page of a dcecp object:        dcecp> man dcecp_<object>
dcecp>
```

- オブジェクトによりサポートされるオペレーションだけを知りたい場合は、*object* **operations** コマンドを使用してください。これによって、オブジェクトに対して行うことのできるアクションのリストが戻されます。次の例は、**principal** オブジェクトについて使用できるオペレーションをリストする方法を示しています。

```
dcecp> principal
operations
catalog create delete modify show operations help
dcecp>
```

  このコマンドを **prin oper** のように省略して、入力の手間を省くこともできます。

- *object* **help** コマンドを使用すれば、オブジェクトおよびそのオペレーションに関するより詳細なヘルプを表示することができます。次の例は、 **principal** オブジェクトによりサポートされる、各オペレーションごとに 1 行の説明を戻します。

```
dcecp> principal help
catalog             Returns all the names of principals in the registry.
create              Creates a DCE principal.
delete              Deletes a principal from the registry.
modify              Changes the information about a principal.
rename              Renames the specified principal.
show                Returns the attributes of a principal.
help                Prints a summary of command-line options.
operations          Returns a list of the valid operations for this command.
dcecp>
```

- *object* **help** コマンドに *operation* 引き数を追加することによって、使用可能なコマンド・オプションに関する情報を表示することができます。次の例は、**principal create** オブジェクトによってサポートされる、各オプションごとに 1 行の説明を戻します。

```
dcecp> principal help create
-alias              Indicates the principal name is an alias of the uid.
-attribute          Specify principal attributes in an attribute list format.
-fullname           Fullname of the principal.
-quota              How many registry objects can the principal create.
-uid                User Identifier of the new principal.
-uuid               Orphaned UUID to be adopted by the principal.
dcecp>
```

- *object* **help -verbose** コマンドを使用することによって、オブジェクト自身に関するヘルプを表示することができます。次の例では、 **principal** オブジェクトの説明とそのオブジェクトの使用法に関する情報とが戻されます。

```
dcecp> principal help -verbose
This object allows manipulation of principal information stored
in the DCE registry.  The argument is a list of either relative or
fully-qualified principal names.  Specify fixed attributes using
attribute options or an attribute list.  Specify any extended attributes
using an attribute list.  Principal operations connect to a registry that
can service the request.  Specify a particular registry by setting the
_s(sec) convenience variable to be a cell-relative or global replica
name, or the binding of the host where the replica exists.  The
completed operation sets the _b(sec) convenience variable to the name
of the registry contacted.
dcecp>
```

- 最後に、ある種の POSIX スタイルのシステムの中には、 Tcl 要約参照ページとともに **dcecp** オブジェクトの参照もできるシステムがあります。各 **dcecp** オブジェクトに、利用可能なオブジェクトおよびオペレーションを説明する独自の参照ページがあります。 **dcecp** オブジェクト参照ページを表示するための一般的な構文は、次のとおりです。

  man *object_name*

  次の例は、principal オブジェクトのための参照ページを呼び出す方法を示しています。 **dcecp** 内から man コマンドを使用できることに注意してください。

```
dcecp> man principal
 .
 . [output omitted]
 .
dcecp>
```

  Tcl 参照ページは、Tcl 組み込みコマンドを要約しています。次のように入力すると、 UNIX スタイルのシステムで Tcl 要約参照ページを表示することができます。

```
dcecp> man Tcl
 .
 . [output omitted]
 .
dcecp>
```

dcecp セッションのカスタマイズ

DCE 制御プログラムには、日常の DCE 管理オペレーションの大部分を実行するための、多くのコマンド、オブジェクト、およびタスク・スクリプトが含まれています。しかし、**dcecp** インターフェースの使用に慣れてくると、新しいコマンドや機能の追加、または既存のコマンドや機能のカスタマイズを行いたくなることがあります。以下の項では、スクリプトや新しいオブジェクトを **dcecp** セッションに追加する方法について説明します。オブジェクトは、 **dcecp** ヘルプ・システムを使用し、*object operation* という形式をとる、スクリプトの公式な実装です。 21ページの『第2章 DCE 制御プログラム・コマンド言語の使用』と 53ページの『第3章 スクリプトおよび dcecp オブジェクトの作成』には、 **dcecp** スクリプトの作成と、新しいオブジェクトの作成について説明してあります。

dcecp セッションへのスクリプトの追加

スクリプトを作成した後、ホストにログインした 1 人または全員にそれを使用できるようにするには、 **dcecp** の初期設定時に呼び出される、次に示すファイルの 1 つまたは複数を変更します。

[info library]/init.tcl
: このファイルが最初に読み取られるファイルであり、これにはホスト用の標準 Tcl 初期化コマンドが入っています。このファイルは、ホストで実行される **dcecp** のすべてのインスタンスに影響を与えます。このファイルには、Tcl **unknown** コマンドと、すべての **dcecp** オブジェクトの初期化に使用される **auto_load** 機能の定義が含まれています。このファイルに **dcecp** カスタマイズを追加するのは避けてください。

dcelocal/**init.dcecp**
: このファイルには、ホストのための **dcecp** 固有の起動情報が入っています。このファイルは、ホストで実行される **dcecp** のすべてのインスタンスに影響を与えます。オペレーションとタスクを実装する **dcecp** スクリプトは、 *dcelocal*/**dcecp** ディレクトリーに格納されています。このファイルにプロシージャー形式でカスタマイズを追加して、それをホスト上のすべての **dcecp** ユーザーが使用できるようにします。

$HOME/.dcecprc
: このオプション・ファイルは、各 **dcecp** ユーザー (**.dcecprc** ファイルの所有者) に影響を与えるユーザー・カスタマイズを格納します。各 DCE ユーザーは、**.dcecprc** ファイルを保守して、私用プロシージャーまたはオペレーションの別名を保管することができます。 **.dcecprc** ファイルを変更することによって、複数の管理者が各環境で柔軟に管理を行うことができます。たとえば、管理者ごとに異なる **.dcecprc** ファイルで **dcecp source** コマンドを使用して、特定の管理領域に合わせて調整された、特定のコマンドおよびタスク・スクリプトを呼び出すことができます。

この項の残りの部分では、簡単なタスク・スクリプトと、スクリプトを個人的に使用するための 1 つの方法を示します。この例は、現在の時刻を示す、制御プログラムの既存の **clock** オブジェクトから開始します。しかし、時刻は、次に示すような、ローカル・ホスト上のクロックからの DTS のタイム・スタンプにすぎません。

```
dcecp> clock show
1994-10-03-10:22:59.991-04:00I-----
dcecp>
```

DTS サーバーからタイム・スタンプを入手するだけでなく、時刻とともに DTS サーバーの名前も表示するプロシージャーを作成するとします。このユーザー作成のプロシージャーの名前は、次の例で呼び出しているように、**show_clock** です。

```
dcecp> show_clock
Time on mars is        1994-09-30-15:03:43.979-04:00I-----
dcecp>
```

このプロシージャーを 1 人のユーザーが使用できるようにするには、そのユーザーの **.dcecprc** ファイルにプロシージャーを組み込みます。次の例の **.dcecprc** ファイルには、 **_dcp_show_clocks** プロシージャーと、より簡単な show_clocks コマンド名を使用してこのプロシージャーを呼び出せるようにする、別名から構成された、ユーザー・カスタマイズが含まれています。 **_dcp_whoami** という名前のもう 1 つのプロシージャーは、現在のログイン識別情報を表示します。 **.dcecprc** ファイル内のオペレーションの順序に注意してください。各プロシージャーはファイルの先頭で定義されています。プロシージャーの名前変更と呼び出しは、プロシージャーの定義後に実行する必要があります。

```
##
## Start up commands
##
# A simple command to rerun .dcecprc after modifications
proc .d {} {source $HOME/.dcecprc}

# Show your current login name and your current cell name.
proc _dcp_whoami {} {
  global _c _u
  return "You are '$_u' logged into '$_c'."
}
# Show the time on all of the dts servers running in your cell.
proc _dcp_show_clocks {} {
    set x [directory list /.:/hosts]
    foreach n $x {
        if {[catch {object show $n/dts-entity}] == 0} {
            set index [string last "/" $n]
            set y [string range $n [incr index] end]
            if {[catch {clock show $n/dts-entity} msg] == 0} {
                set i [expr 20 - [string length $y]]
                puts [format "Time on $y is %${i}s %s" " " ¥
                    [clock show $n/dts-entity]]
            } else {
                set i [expr 20 - [string length $y]]
                puts [format "Time on $y is %${i}s %s" " " ¥
                   "Server not responding."]
            }
        }
    }
}

# Give some procs usable names
rename _dcp_whoami whoami
rename _dcp_show_clocks show_clocks

# If I am authorized, say so
if {$_u != ""} {
  whoami
}
```

ファイルの終わり近くの **rename** コマンドのおかげで、より簡単なコマンド名 **show_clocks** と **whoami** でもって **_dcp_show_clocks** と **_dcp_whoami** のプロシージャーを呼び出すことが可能となります。

dcecp プログラムを開始すると、DCE にログインしたときに、このファイルの最後の部分によって **_dcp_whoami** のプロシージャーが呼び出されます。 **_u** コンビニエンス変数が設定してあれば、 **_dcp_whoami** プロシージャーによって、次のように現在のログイン識別情報が印刷されます。

```
% dcecp
You are 'principal_name' logged into 'cell_name'.
dcecp>
```

DCE 制御プログラムへの新しいオブジェクトの追加

スクリプトを公式の **dcecp** オブジェクトとして作成しておくと、他のタスク・オブジェクトが存在している同じディレクトリーに新しいオブジェクトを組み込むことによって、それを使用可能にすることができます。 UNIX システムでは、多くの場合、これは *dcelocal***/dcecp** です。規則として、DCE セル内の各ホストに新しいオブジェクトを追加するようにしてください。 53ページの『第3章 スクリプトおよび dcecp オブジェクトの作成』には、**dcecp hostdata** オブジェクトを使用してスクリプトあるいは他のファイルをセル内の各ホストにコピーする方法の説明があります。

新しいスクリプトをインストールする場合、**auto_mkindex** ユーティリティーを実行して、ホスト上の他のユーザーがその新しいオブジェクトを使用できるようにする必要があります。 **auto_mkindex** ユーティリティー実行の詳細については、 53ページの『第3章 スクリプトおよび dcecp オブジェクトの作成』を参照してください。

環境変数

環境変数とは、ユーザー自身が設定できる、 DCE により使用される変数です。 DCE 環境変数全般については、 *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* を参照してください。

第2章 DCE 制御プログラム・コマンド言語の使用

3ページの『第1章 DCE 制御プログラムの紹介』では、 DCE 制御プログラムを使用して DCE 環境を管理するいくつかの方法をハイレベルな観点から見てきました。この章では、 **dcecp** 管理コマンドとタスク・スクリプトを構成するために必要な構文規則と、いくつかのより重要なコマンドについて説明します。

dcecp コマンド言語は、 **directory create** や **object modify** などの DCE 管理コマンドと、 **if** や **foreach** などの Tcl 組み込みコマンドから成っています。ここでは、DCE 管理コマンドの説明は行いません。これらについては、特定の DCE コンポーネントの管理を論じるセクションで説明します。その代わりに、汎用性の高い構文規則や組み込みコマンドの使用法に焦点をあてます。

dcecp の基本となるツール・コマンド言語 (Tcl) は、 **dcecp** 以外の他のアプリケーションにも使用される汎用言語です。 Tcl をいろいろな目的に使用する方法はいくつもありますが、ここでの説明は、DCE 環境の管理に使用される可能性が最も高いコマンドに限定します。さらに、このコマンドの説明は、個々のコマンドのすべての局面を説明するものではありません。むしろ、DCE 環境を管理する状況でコマンドを使用する理由とその方法を提案します。まだ Tcl に慣れていない場合は、おそらく、適切なコマンドとタスク・スクリプトを作成するために、 Tcl 参照ページを含め、適切な Tcl 資料を利用する必要があります。

この章の概要

この章では、基本的な **dcecp** 構文規則について説明した後、対話式コマンドおよびタスク・スクリプトで使用するいくつかのコマンドについて見てみます。ここでの説明は、以下を中心にします。

- コマンドまたはスクリプト内でデータを簡単に渡す方法としての変数の使用
- あるコマンドの出力を別のコマンドの入力として渡す方法としてのコマンドの置換
- **dcecp** がコマンドを正しく構文解析するための要素のグループ化
- 情報を分類、検出、および再利用するためのリストの使用
- コマンドおよびタスク・スクリプトにおける算術関数の使用
- **if** ステートメントとループによるスクリプトの条件付けと制御
- **case** コマンドの使用による文字パターン関連スクリプトの実行
- **eval** の使用によるコマンドの統合
- **source** によるオペレーションのインポート
- **proc** による新しい **dcecp** コマンドの作成
- エラーおよび例外情報の使用法
- ストリングの取り扱い
- ファイルの処理
- サブプロセスの作成

変数の置換

他のプログラミング言語と同様に、 **dcecp** には値を表現し使用するための簡略法があります。変数置換はこの簡略法の 1 つであり、これを使用すると、値 (たとえば、 CDS ディレクトリー内のオブジェクトの名前) を変数として表示することができます。

変数の値を設定するためには、**set** コマンドを使用してください。読みやすくするために、変数名は文字、数字、および下線文字の任意の組み合わせで構成することができます。スペースを変数名に組み込む (これは通常はお勧めできません) または値に組み込む場合には、 "" (引用符) または ¥ (円記号) を使用してください。以下の例はすべて、有効な変数名を使用しています。

```
set a $i
set CDS_clearinghouse_name cambridge_ch
set DCE_user_1 "William Rosenberry"
```

次の例は、変数 *a* に値 **7** を設定するものです。 2 番目の例のように値を指定せずに **set a** コマンドを使用すると、 **dcecp** は変数の現在の値を表示します。

```
dcecp> set a 7
7
dcecp> set a
7
```

dcecp set コマンドを使用して変数に値を設定しておくと、スクリプトまたは対話式コマンド内のその他の場所で、今後この変数を使用することができます。 DCE 制御プログラムでは、コマンド・ワードに現行値を挿入させるために $ (ドル記号) を使用します。簡単な例を次に挙げます。

```
dcecp> set a 7
7
dcecp> expr $a+2
9
```

この最初の例では、変数 *a* は 7 に設定されます。 2 行目では、**expr** コマンドで *a* の値 (7) に 2 を加えています。ドル記号によって、 **dcecp** が値 7 を挿入させます。最後の行は、 **expr** コマンドから戻された値を示しています。

さらに適切な例は次のとおりです。

```
dcecp> set a /.:/sec
/.:/sec
dcecp> object show $a
{RPC_ClassVersion
 {01 00}}
{RPC_ObjectUUIDs
 {06 3b 23 00 72 e5 e0 1d 8c b4 00 00 c0 8a df 56}}
{RPC_Group
 {2f 2e 2e 2e 2f 77 61 72 64 5f 63 65 6c 2e 6f 73 66 2e 6f 72
 67 2f 73 75 62 73 79 73 2f 64 63 65 2f 73 65 63 2f 6d 61 73 74
 65 72 00}}
{CDS_CTS 1994-05-23-17:21:37.481+00:00I0.000/00-00-c0-8a-df-56}
{CDS_UTS 1994-05-23-17:22:36.607+00:00I0.000/00-00-c0-8a-df-56}
{CDS_Class RPC_Group}
{CDS_ClassVersion 1.0}
dcecp>
```

次のように、**unset** コマンドを使用すると、変数を削除することができます。

```
dcecp> unset a
dcecp> set a
Error: cannot read "a": no such variable
dcecp>
```

コマンド置換

コマンド置換は、あるコマンドの戻り値を別のコマンド内に表示する便利な方法を提供します。これは、あるコマンドの戻り値を別のコマンドへの入力として使用したい場合に便利です。コマンド置換を起動するには大括弧を使用してください。次の例は、**expr** コマンドを使用しています。これについて簡単に説明します。一般に、**expr** は算術関数を実行し、その引き数によって表される算出値を戻します。

```
dcecp> set a 4
4
dcecp> set b [expr $a+2]
6
dcecp> set b
6
dcecp>
```

もっと実際的な例では、長い名前もしくはリストを戻すコマンドに対してコマンド置換を使用する場合があります。 3ページの『第1章 DCE 制御プログラムの紹介』で見た例を思い出してください。この例では、 **[group list temps]** コマンドが **foreach** コマンドにリストを戻し、そのリスト内の各要素に対して **account modify** オペレーションが行われます。この節では、**foreach** ループ・コマンドについてもっと詳しく説明します。

```
dcecp> foreach i [group list users] {
> account modify $i -change {expdate 1995-12-31}}
dcecp>
```

コマンド置換のもう 1 つの実際的な使用法は、 **if** ステートメントのためのテスト条件を設定することです。この使用法の例は、36ページの『if ステートメントによる条件付け』に示してあります。

要素のグループ化と解釈の制御

プログラム言語は、多くの場合、個々の要素に対してではなく、グループとして選択された要素に働きかけるために、中括弧、引用符、小括弧などの記号を使用します。 **dcecp** も、"" (引用符) と {} (中括弧) を使用して、要素を構造体としてグループ化します。二重引用符を使用すると、通常は別個に構文解析される要素をグループ化し、単一要素として処理されるようにすることができます。 **dcecp** がコマンドおよびその他のデータ (たとえば、戻り値) を正しく解析できるように、要素をリストにグループ化するためには、中括弧を使用します。

dcecp コマンドの要素を区切るには、ホワイト・スペース、すなわち、スペース文字、タブ文字、および改行文字を使用します。次の **dcecp** コマンドでは、スペース文字を使用して 3 つの要素を区切っています。

```
dcecp> directory create /.:/subsys/comm_services
dcecp>
```

スクリプト内のコマンドを区切るには、改行文字か ; (セミコロン) のいずれかを使用します。以下の 2 つの例は変数を設定してからそれを使用するものですが、これらは同じです。

```
dcecp> set a /.:/subsys/comm_services
/.:/subsys/comm_services
dcecp> directory create $a
dcecp>

dcecp> set a /.:/subsys/comm_services; directory create $a
dcecp>
```

要素のグループ化に中括弧を使用するか、引用符を使用するかは、 $、[、および { のような特殊文字を **dcecp** にどのように解釈させるかによって決まります。中括弧はこうした文字の大部分の特殊な解釈を使用不可にしますが、二重引用符が特殊な解釈を使用不可にするのはほんの数文字です。円記号は、特殊文字の解釈を使用不能にするもう 1 つの方法です。これについては、25ページの『円記号による特殊文字の取り込み』で説明します。中括弧、引用符、および円記号を一緒に使用すると、 **dcecp** コマンド・ストリングの構成に多くの柔軟性が得られます。

中括弧による要素のグループ化

中括弧は、別々の要素を中括弧でグループ化して、左括弧 ({) とそれに対応する右括弧 (}) との間にあるすべてのものから構成される、新しい要素を作成します。また、中括弧で囲まれた要素をネストすることもできます。以下の例の各リストには 3 つの要素が含まれています。

```
larry moe curly

1 {3 5 7 11 13} {17 19}

red {orange yellow {green blue} indigo} violet
```

中括弧は、コマンド ([ ])、変数 ($)、および円記号の置換を不能にします。中括弧の最も重要な用法は、 **dcecp** コマンドに正しい数の引き数があることを確認することですが、これは、リストに特殊文字を組み込む便利な方法も備えています。これがどのように作動するかを調べるために、次の例を考えてみます。

```
dcecp> set a solution
solution
dcecp> puts $a
solution
dcecp> puts {This is a convenient $a}
This is a convenient $a
```

puts コマンドは、しばしばファイルへの書き込みに使用されますが、引き数を 1 つだけ指定して呼び出された場合は、その引き数を **stdout** に書き出します。上記の例に戻ると、最初の **puts** では、変数 *a* は通常どおりに解釈されます。 2 番目の **puts** では、スペース文字とドル記号の特殊解釈を使用不能にすることによって、個別の要素が 1 つの引き数にグループ化されます。

二重引用符による要素のグループ化

中括弧と同じように、二重引用符も要素をグループ化します。ただし、中括弧の場合と異なり、二重引用符をネストすることはできません。さらに、中括弧はほぼすべての特殊文字を使用不能にしますが、二重引用符が使用不能にするのは、数個の特殊文字だけ、すなわち、スペース、タブ、改行文字およびセミコロンだけであ

り、そのため、テキスト要素のストリング内での円記号の不便な使用を避けることができます。二重引用符の最も便利な使用法は、ドル記号のトリガーを使用して変数を読み取りやすく拡張できるようにすることです。たとえば、次の例では、変数 (*a*) を、スペースを含んだ値に設定します。

```
dcecp> set a "XYZ server
for /.:/corp/comm_groups"
XYZ server for /.:/corp/comm_groups
dcecp> puts $a
XYZ server for /.:/corp/comm_groups
dcecp>
```

二重引用符を使用しても、コマンド、変数、および円記号の置換は使用不能にはなりません。 24ページの『中括弧による要素のグループ化』で使用した例を、少し変化させてみます。

```
dcecp> set
a solution
solution
dcecp> puts $a
solution
dcecp> puts "This is a convenient $a."
This is a convenient solution.
dcecp>
```

この例では、2 番目の **puts** コマンドで引用符を使用し、スペース文字の特殊解釈を不能にすることによって、 5 つの要素を **puts** のための 1 つの引き数にまとめています。ただし、引用符は、ドル記号の解釈には影響を与えません。

円記号による特殊文字の取り込み

dcecp が、スペース、中括弧、引用符、またはドル記号などの特定の特殊文字を使用して要素の解釈を制御することについては、これまでに説明しました。場合によっては、特殊解釈を一時的に中断して、ストリングに特殊文字を 1 つ指定することが必要になる場合があります。円記号は、その直後の文字の特殊解釈を抑止する、一種の置換を行います。

要素のストリングに印刷しないスペース文字を挿入するには、円記号を使用します。たとえば、以下の **dcecp** リストには 3 つの要素があります。

```
  a b¥ c d
  a b ¥{
```

最初の例の要素は、**a**、**b c**、および **d** です。 2 番目の例の要素は、**a**、**b** および **{** です。もっと実用性の高い例としては、以下のコードの断片のように、円記号を使用してエラー・メッセージに引用符を組み込むことができます。

```
if {[llength $a] < 2} {
   error "Unable to parse ¥"$element_list¥"."
}
```

次のリストは、円記号を使用して要素のストリング内に組み込むことができる特殊文字を示しています。

¥b	後退
¥t	タブ
¥e	エスケープ

¥n	改行
¥r	復帰
¥{	左中括弧
¥}	右中括弧
¥[	左大括弧
¥]	右大括弧
¥$	ドル記号
¥ (スペース)	スペース (″ ″)
¥;	セミコロン
¥″	二重引用符
¥¥	円記号
¥(改行)	なし
¥*ddd*	8 進値

コメントによるスクリプトの文書化

スクリプトを作成する場合、スクリプトの働きを自分自身や他のユーザーに知らせるためにコメント行を指定したい場合があります。コメントを挿入するためには、# (番号記号) を使用します。 DCE 制御プログラムは、番号記号と次の改行との間の解釈を抑止します。番号記号は、**dcecp** がコマンドの先頭文字を期待する場所に指定する必要があります。次の例はいずれも有効です。

```
set a 5
# sets a to 5

set a 5 ;# sets a to 5
```

次の例の番号記号は、 **dcecp** がコマンドの先頭文字を期待する場所以外の所にあるため、無効です。

```
set a 5 # sets a to 5
```

コメントの一般的な用途は、次の例のスクリプトの断片のように、スクリプト内のプロシージャーを文書化することです。

```
#
# _dcp_cleanup_user_create  - This function undoes changes
# after a failure in one of the user create functions as
# though the operation never occurred.
#

proc _dcp_cleanup_user_create {account_name args}
{
```

コンビニエンス変数

DCE 制御プログラムは、コマンド出力だけでなく、入力された情報も記憶し、情報の特定の部分を、以後のコマンドで再利用するために、コンビニエンス変数に保管します。対話式コマンドでこのような変数を使用することによって、入力の手間を省き、入力ミスを減らすことができます。

コンビニエンス変数が適用されるのは、 **directory**、**principal**、**acl**、 **account** などの **dcecp** コマンドだけです。 **for** または **eval** などの Tcl コマンド、あるいは **mv** または **grep** などの UNIX コマンドには適用されません。 1 例として、コンビニエンス変数 **_n** は、次のような **principal create** オペレーションで使用される名前 (引き数) を保持します。 **principal show** オペレーションが、 **$_n** 変数を使用して、その名前を検索します。

```
dcecp> principal create D_Kalivas
dcecp> principal show $_n -all
{fullname {}}
{uid 17}
{uuid 00000011-d957-21cd-8d00-0000c08adf56}
{alias no}
{quota unlimited}
dcecp>
```

この単純な説明は、コンビニエンス変数の一般的なオペレーションを示していますが、この変数の便利さを十分に示していません。大部分のコンビニエンス変数は、対話モードでの使用を補助することを目的としていますが、一部のコンビニエンス変数はスクリプトでも使用することができ、変数に含まれる情報がスクリプト内にハードコーディングされていないため、柔軟性が増します。さらに、DCE 制御プログラムの経験を積むと、これらの変数が必要不可欠な管理ツールであることに気付きます。

DCE 制御プログラムには、以前に入力した情報またはコマンド出力と置き換わる、複数のコンビニエンス変数が用意されています。コンビニエンス変数は、どれも _ (下線) で開始され、 1 文字の変数名を他の目的に使用することができます。

以下の項では、コンビニエンス変数について説明します。説明する順序は、一般的に、似ている変数または関連する変数をまとめています。

現行プリンシパル (ユーザー) 名 (_u)

_u コンビニエンス変数は、現行の単純プリンシパル名を保持します。 DCE 制御プログラムは、親プロセスから継承したログイン・コンテキストから、この変数を設定します。この値を変更するには、別の **login** オペレーションを実行します。この値を **set** によって設定すると、エラーになります。

```
dcecp> puts $_u
cell_admin
dcecp>
```

この変数は、特定の DCE 識別情報かどうかを、次に進む前にテストする、スクリプト内で実際には使用されます。正しくない識別情報を検出すると、スクリプトは、必要な識別情報を入力するようにプロンプトを出して、 **dce_login** オペレーションを実行します。

完全修飾プリンシパル名の作成については、 『現行セル名 (_c)』の項のセル名変数の説明を参照してください。

現行セル名 (_c)

_c コンビニエンス変数は、プリンシパルが登録されているセルの名前を保持します。 DCE 制御プログラムは、親プロセスから継承したログイン・コンテキストか

ら、この変数を設定します。この値を変更するには、別の **login** オペレーションを実行します。この値を **set** によって設定すると、エラーになります。

```
dcecp> puts $_c
/.../my_cell.goodco.com
dcecp>
```

この変数は、一般的に、管理者が複数のセルを扱う環境で便利です。たとえば、スクリプトで使用するために現行コンテキストの完全修飾プリンシパル名を構成するときに、構築ブロックとして **_c** 変数を使用することができます。次の例に示すように、セル名変数とユーザー名変数を斜線文字 (/) で結合してください。

```
dcecp> puts $_c/$_u
/.../my_cell.goodco.com/cell_admin
dcecp>
```

現行ホスト名 (_h)

_h コンビニエンス変数は、現行ホストの DCE 名を保持します。 DCE 制御プログラムは、 **dcecp** の起動時にこの変数を設定します。この値を **set** によって設定すると、エラーになります。

```
dcecp> puts $_h
hosts/planets
dcecp>
```

_h 変数は、ホストの名前を対話式ユーザーに戻す場合に便利です。 **_c** 変数とともにこれを使用して、次に示すように、ホスト・プリンシパル名などの名前をスクリプト内に構成することもできます。

```
dcecp> puts $_c/$_h/self
/.../my_cell.goodco.com/hosts/planets/self
dcecp>
```

最新オペレーションの引き数名 (_n)

_n 変数は、最新の制御プログラム・オペレーションの引き数として使用された名前 (単数または複数) を保持します。ほとんどの **dcecp** オブジェクトは、引き数として、名前または名前のリストを使用します。引き数に名前を使用しないオブジェクトには、各種の **dcecp** コマンド、**dcecp_initInterp**、 **login**、**logout**、**errtext**、**quit**、 **resolve** および **shell** があります。

名前は、通常、次の **directory** オペレーションで示すように、 **dcecp** オペレーションの 3 番目の引き数です。

```
dcecp> directory create /.:/sales/printers/text_printers
dcecp>
```

$_n は、一度設定されれば、以後のオペレーションで名前引き数の代わりに使用することができます。たとえば、上記の例で作成した **/.:/sales/printers/text_printers** ディレクトリーのディレクトリー属性を次のようにして変更することができます。

```
dcecp> directory mod $_n -change {CDS_Convergence
low}
dcecp>
```

_n 変数は、 1 つのディレクトリー・サービス・オペレーションを複数の名前について実行する場合などは、名前のリストを保持することもできます。たとえば、複数のディレクトリーを作成後、属性を変更することができます。

```
dcecp> directory create {
> /.:/sales/printers/text_printers
> /.:/sales/printers/graphics_printers
> /.:/sales/printers/colorgraphics_printers }
dcecp>
```

以後のディレクトリー・サービス・オペレーションでは、名前または名前のリストの代わりに、 **_n** 変数を使用するだけで済みます。

```
dcecp> directory
modify $_n -change {CDS_convergence high}
dcecp>
```

_n の親 (_p)

_p 変数は、**_n** に保管されている名前の親を保持します。 **_n** 変数は、最新のオペレーションの引き数の中で使用された名前または名前のリストを保持します (28ページの『最新オペレーションの引き数名 (_n)』を参照)。 **_p** 変数は、 **_n** 内の名前より階層内で上の (セル・ルートにより近い) 名前または名前のリストを保持します。

_p 変数の 1 つの使用法は、ディレクトリーの CDS 階層を上方に向かって調べることです。もう 1 つの使用法は、親オブジェクトの*アクセス制御リスト* (ACL) を表示することです。次の操作は、サーバー構成オブジェクトとその親オブジェクト (**/.:/hosts/krypton/config/srvrconf**) の ACL を表示します。

```
dcecp> acl show /.:/hosts/krypton/config/srvrconf/video_clip
{appl_admin cdfrwx}
{unauthenticated r}
{any_other r}
dcecp>
dcecp> puts $_p
/.:/hosts/krypton/config/srvrconf
dcecp>
dcecp> acl show $_p
{appl_admin criI}
{unauthenticated r}
{any_other r}
dcecp>
```

最後の dcecp オブジェクト名 (_o)

_o 変数は、最新のオペレーションで使用された **dcecp** オブジェクトの名前を保持します。次の例は、**account** を再入力するのを避けるために、 **_o** 変数を使用しています。

```
dcecp> account show j_wanders
{acctvalid yes}
{client yes}
 .
 .   [output omitted]
 .
{home /}
 .
 .   [output omitted]
 .
{shell {}}
{stdtgtauth yes}
dcecp> $_o modify j_wanders -home /.:/fs/corporate_services/users/j_wanders
dcecp>
```

最後のオペレーションの戻り値 (_r)

_r 変数は、最新のオペレーションの戻り値を保持します。多くの **dcecp** コマンドは、リスト形式の複数行からなる出力を戻します。

_r コンビニエンス変数の使用例を次に示します。 **dts show** コマンドは、複数行をリスト形式で戻します。 **attrlist getvalues** オペレーション (**attrlist(8dce)** 該当ページを参照) は、戻されたリストを探索してストリング **toofewservers** を見つけ出し、そしてその関連する値を戻します。

```
dcecp> dts show -counters
{creationtime 1994-09-16-07:50:13.067-04:00I-----}
{nointersections 0}
{nointersections 0}
{diffepochs 0}
{toofewservers 1}
{providertimeouts 82}
{badprotocols 0}
{badtimerep 0}
{noglobals 81}
{noresponses 0}
{abrupts 0}
{epochchanges 0}
{syserrors 0}
{syncs 1574}
{updates 0}
{enables 1}
{disables 0}
{nomemories 0}
{providerfailures 0}
{badlocalservers 0}
{badservers 0}
dcecp> attrlist getvalues $_r -type toofewservers
1
dcecp>
```

使用する DCE サーバー (_s(xxx))

_s(*xxx*) 変数は、次の DCE オペレーションで使用される DCE サーバーの名前を保持します。 DCE 制御プログラムには、このような変数が 4 つ用意されています。これらの変数は **dcecp** によっては設定されないので、使用したい場合には、ユーザーが設定しなければなりません。変数には、次のものがあります。

_s(sec)

この変数は、次のレジストリー・オペレーションに使用したいセキュリティー・サーバーの名前を保持します。読取専用レプリカを指定するためにこの変数を設定し、しかもオペレーション (たとえば **principal create**) にマスター・レプリカが必要な場合は、 **dcecp** は、変数を無視して、マスター・レジストリーにバインドしようとします。 **_s(sec)** 変数を使用するレジストリー・オペレーションには、 **principal**、**group**、**organization**、**registry**、 **account** および **xattrschema** があります。

DCE 制御プログラム・オペレーションは、**_s(sec)** 変数を、使用された最新のレジストリーの名前を保持している **_b(sec)** 変数とともに使用します。 **registry** オペレーションは、以下の順序でセキュリティー・サーバーを選択します。

1. **registry** オペレーションに名前引き数として渡されたサーバー

2. オペレーションに名前引き数がない場合は、 **_s(sec)** 変数に指定されているサーバー
3. **_s(sec)** 変数が設定されていない場合は、 **_b(sec)** 変数に指名されているサーバー
4. **_b(sec)** 変数が設定されていない場合 (すなわち、これが **dcecp** プログラムの初期設定後の最初の **registry** オペレーションである場合) は、サービスがオペレーションに適切なサーバーを任意に提供する。

_s(cds)

この変数は、次のディレクトリー・サービス・オペレーションに使用したい CDS サーバーの名前を保持します。この変数を設定すると、CDS オペレーションは、指定されたサーバーの使用を試みます。この試みが、サーバーが何らかの理由で使用できないなどが原因で失敗すると、オペレーションも失敗します。このような失敗に対処するには、この変数を **unset** するか、またはサーバーを使用可能にする必要があります。

アプリケーションのすべての要件が、この変数の中で指定されたクリアリングハウスで満たされる場合には、 **_s(cds)** 変数を使用するのが賢明です。CDS 内の名前の探索が、複数のクリアリングハウス内のディレクトリーに渡る可能性がある場合には、 **_s(cds)** 変数を使用しないでください。このような場合、**_s(cds)** 変数を指定すると、指定されたクリアリングハウスに検索オペレーションの対象が限定されるので、検索エラーになる可能性があります。

_s(dts)

この変数は、次のタイム・サービス・オペレーションに使用したい DTS サーバーの名前を保持します。この変数を設定すると、DTS オペレーションは、指定されたサーバーの使用を試みます。この試みが、サーバーが何らかの理由で使用できないなどが原因で失敗すると、オペレーションも失敗します。このような失敗に対処するには、この変数を **unset** するか、またはサーバーを使用可能にする必要があります。

この変数の 1 つの使用法は、モニタリング目的のために DTS オペレーションを単一の DTS サーバーに限定することです。通常、タイム・サービス・オペレーションは、どの使用可能な DTS サーバーも使用することができます。

_s(aud)

この変数は、監査オペレーションに使用したい監査デーモンの名前を保持します。デフォルトでは、監査オペレーションはローカル・ホストの監査デーモンを使用します。リモート・ホストの監査デーモンを操作するには、次のように、その名前を **_s(aud)** 変数の値として指定します。

```
dcecp> set _s(aud) /.:/hosts/planets/audit-server
/.:/hosts/planets/audit-server
dcecp>
```

_s(aud) を設定すると、監査オペレーションは、指定された監査デーモンの使用を試みます。この試みが、指定された監査デーモンが何らかの理由で使用できないなどが原因で失敗すると、オペレーションも失敗します。このような失敗に対処するには、この変数を **unset** するか、または当該監査デーモンを使用可能にする必要があります。

DCE サーバーまたは監査デーモンは、以下のいずれとして指定することもできます。

- DCE 名。グローバル・レジストリー名の例を挙げれば、**/.../my_cell.goodco.com/subsys/dce/sec/oddball** です。セルに相対的な CDS クリアリングハウス名の例は、 **/.:/Paris_CH** です。
- サーバーが常駐しているホストに関するストリング・バインディング。ストリング・バインディングは、セキュリティー・サーバー、 DTS サーバー、および監査デーモンを表すことができます。 CDS サーバーを表すことはできません。ストリング・バインディングの例としては、 **{ncacn_ip_tcp 110.15.22.131}** があります。 DCE 制御プログラムは、バインディングを解決して、ホスト上の適切なサービスにします。
- セルの名前。リモート・セルの場合、 **/.../my_cell.goodco.com** のようなグローバル・セル名を指定してください。ローカル・セルの場合は、ルートを **/.:** と指定することができます。これらのオペレーションは、オペレーションに適切な任意のサーバーを使用します。

最後に使用されたセキュリティー・サーバー (_b(sec))

_b(sec) コンビニエンス変数は、最新の **registry** オペレーションで使用されたセキュリティー・サーバーの名前を保持します。 DCE 制御プログラムは、直前のレジストリー・オペレーションに基づいてこの変数を設定します。したがって、ユーザーはこの変数を見るだけで、設定することはできません。

この変数の値を読む 1 つの理由は、次の例に示すように、どのレジストリーが最新のオペレーションを実行したかをチェックするためです。

```
dcecp> puts $_b(sec)
/.../my_cell.goodco.com/subsys/dce/sec/oddball
dcecp>
```

レジストリー・オペレーションは、 **_b(sec)** 変数の値を、**_s(sec)** 変数の値とともに使用して、使用すべきセキュリティー・サーバーを決定します。 **_s(sec)** 変数、およびこれらの値をレジストリー・オペレーションに使用する方法については、 30 ページの『使用する DCE サーバー (_s(xxx))』を参照してください。

最新のエラー・コード (_e)

_e コンビニエンス変数は、検出された最後の DCE エラー・コードを保持します。 DCE 制御プログラムがエラー・コードを判別できた場合に、この変数が設定されます。実際のエラー・コードが不明の場合は、変数は **-1** (負数の 1) に設定されます。

CDS 信用レベル (_conf)

_conf コンビニエンス変数は、要求を満たすローカル CDS デーモンの信用度を指示します。これは、CDS オブジェクトに対して働くほとんどのコマンドの動作を変更します。信用レベルには、 **low**、**medium**、および **high** があります。

式による評価とカウント

expr コマンドは、スクリプト内の算術関数を柔軟に表現し使用する方法を提供します。式は、リスト内の要素数などの数値情報の比較、モニター目的のためのしきい値の設定、スクリプトの実行を制御するカウンターの増分、および統計情報の作成などに便利です。

単純な **dcecp** 式は、 **+** (加算) または ***** (乗算) などの演算子といくつかのオペランドの組み合わせです。 **expr** コマンドは、式を 1 つの引き数としてとるので、式にスペースを入れる場合は、括弧または中括弧が必要になります。式の中でのグループ化を制御する場合には括弧を使用してください。また、式をネストすることもできます。以下の式はすべて有効です。

```
dcecp> expr {2 + 3}
5
dcecp> expr 2+3
5
dcecp> set x 24
24
dcecp> expr ($x-8)*2
32
dcecp> expr $x-(8*2)
8
dcecp> expr $x-8*2
8
dcecp>
```

式の中で変数を使用する場合は注意してください。上記の **$x** のような変数は、4*6 のような非数値ストリングではなく、 24 のような数値ストリングでなければなりません。

DCE 制御プログラムは、通常、数値を 10 進数形式の整数として扱いますが、 8 進数形式または 16 進数形式の数値を読み取ることもできます。 8 進数解釈の場合は、0477 のように、数値の前に 0 (ゼロ) を置いてください。 16 進数解釈の場合は、 0x9FF のように、数値の前に 0x を置いてください。また、ANSI C 標準で指定されたいずれかの形式 (ただし、 f、F、l、および L 接尾辞を除く) を使用して、数値を浮動小数点形式で表すこともできます。

DCE 制御プログラムでは、式の中で cos、exp、log、tan、sin などの多数の数学関数をサポートします。そのときは、同じ名前の C 数学ライブラリー関数を起動します。

次のリストは、**expr** コマンドで使用できる演算子の一部です。このリストの順序は、優先順位を表します。これは、たとえば、**expr** は加算の前に乗算を行うことを意味します (2+2*4 = 10)。

-	単項減算
~	ビットの NOT
!	論理否定
*****	乗算
/	除算
%	剰余
+	加算

-	減算
<<	左桁送り
>>	右桁送り
<	より小 (ブール演算)
<=	より小または等 (ブール演算)
>	より大 (ブール演算)
>=	より大または等 (ブール演算)
==	等 (ブール演算)
!=	不等 (ブール演算)
&	ビット単位 AND
^	ビットの排他 OR
|	ビットの OR
&&	論理 AND
||	論理 OR
*a***?** *b***:** *c*	条件付きブランチ (C の if-then-else)

リストでの操作

リストは、プリンシパル、グループ・メンバー、またはその他のオブジェクトなどの集合上で操作するための便利な方法を提供します。リストは、ユーザーによって入力されたオブジェクトまたはコマンドから戻されたオブジェクトの集合です。この章の前述の例でリストを参照しましたが、それらのリストは、スペース、タブ、または改行によって区切られる、任意の数の要素です。通常、リストは中括弧で囲まれています。

以下の例はすべてリストです。

```
{n_long l_jones p_sawyer d_witt m_dougherty s_preska}
{{/.:/hosts} {/.:/subsys}}
```

DCE 制御プログラムでは、 **dcecp** コマンド・インタープリターが要素を正しく解析できるように、リストを使用して要素をグループ化します。たとえば、set コマンドは次の 2 つの引き数を取ります。

```
set varName value
```

次の **set** コマンドは、 **dcecp** が 3 番目の引き数を検出するため、正しく構文解析されません。

```
dcecp> set a John Hunter
Error: wrong # args: should be "set varName ?newValue?"
dcecp>
```

次に示すように、中括弧、引用符、または円記号を使用して、有効なリストにしてください。

```
dcecp> set a {John Hunter}
John Hunter
dcecp> set a "John Hunter"
```

```
John Hunter
dcecp> set a John¥ Hunter
John Hunter
dcecp>
```

リスト上で作動するコマンドは、リスト内の個々の要素または要素のグループを評価、選択、および実行するための便利な方法を提供します。 DCE 制御プログラムは、リストの作成、変更、探索、分類、およびリスト相互間の変換を可能にする、包括的なコマンドのセットを提供しています。

たとえば、以下のスクリプトはリスト内の最後の要素を戻します。 **llength** コマンドは、リスト内の要素の数を戻します。このリストには要素が 4 つあるので、**llength** は **4** を戻します。 DCE 制御プログラムは、左から右の順番で要素に **0** から始まる番号を付けます。したがって、要素が 4 つあるこのリストには、 **0**、**1**、**2** および **3** の番号が付けられます。 *c* の値は、リストの最後の要素の番号 (3) に設定されます。最後に、**lindex** コマンドは要素 2 (**f**) を戻します。

```
dcecp> set a {a b {c
d e} f}
a b {c d e} f
dcecp> set b [llength $a]
4
dcecp> set c [expr $b-1]
3
dcecp> lindex $a $c
f
dcecp>
```

DCE 制御プログラムは、リストを処理するための数々のコマンドを備えています。**concat** コマンドを使用すれば、リストを結合することができます。既存のリストに要素を追加するには、 **linsert** を使用してください。ある範囲の要素を抽出するには **lrange** を使用し、リスト内の要素を置き換えるには **lreplace** を使用し、リスト要素をアルファベット (辞書) 順に分類するには **lsort** を使用してください。 DCE 制御プログラムには、 **attrlist** オブジェクトも含まれています (リスト要素の操作での使用については **attrlist(8dce)** を参照してください)。

以下の例は、ツリー内のすべての子ディレクトリーをアルファベット順にリストします。 **_r** は、最後のコマンドの出力を保持する **dcecp** コンビニエンス変数です。この例では、**_r** は、 **directory list -simple** コマンドによって戻されるディレクトリーのリストを保持します。

```
dcecp> directory list -simple /.:
hosts subsys cell-profile fs lan-profile planets_ch sec sec-v1
dcecp> lsort $_r
cell-profile fs hosts lan-profile planets_ch sec sec-v1 subsys
dcecp>
```

スクリプトの制御

DCE 制御プログラムには、スクリプトの実行を制御するための複数のコマンドが備えられています。 **if**、 **while**、**for**、**foreach**、および **case** などのコマンドは、いろいろな条件の下でスクリプトの各部を実行します。 **break** と **continue** は、コマンド・スクリプトの一部または全部の実行を停止させることができます。

if ステートメントによる条件付け

場合によっては、スクリプトの一部を特定の条件でのみ実行させたいことがあります。条件を検出し、条件付きでオペレーションを実行するには、 **if** ステートメントを使用します。 **if** ステートメントの構文は次のとおりです。

```
if test true_body else false_body
```

属性のリストを探索して特定の属性を見つけ出すスクリプトを作成するとします。 **if** ステートメントは、属性が存在するかどうかによって、特定のアクションをとります。次のスクリプトの断片の例では、アカウント名が **list_of_group_entries** 変数内に存在しないと、エラー・メッセージが戻されます。

```
set list_of_group_entries [group list $group -simplename]
if { [lsearch $list_of_group_entries $account_name] == -1} {
     group add $group -member $account_name
} else {
error "Group ¥"$group¥" already has an entry ¥
  for ¥"$account_name¥"."
}
```

ループによるスクリプトの実行の制御

プログラム言語は、指定された条件が存在する限りオペレーションを繰り返すために、ループを使用します。 DCE 制御プログラムは、3 種類のループ、すなわち、**foreach**、**while**、および **for** を提供しています。使用するループのタイプは、条件が指定される方法によって決まります。

foreach ループ

リスト内の各要素に対して特定のオペレーションを実行したい場合は、 **foreach** コマンドを使用します。リストは、オブジェクトの集合、すなわち、ユーザーが入力したかまたはコマンドから戻されたものの集合であることを思い出してください。

構文は次のとおりです。

```
foreach variable_name list body
```

foreach コマンドは、リスト、スクリプト本体、およびリストの各要素を表す変数で構成されます。このコマンドは、変数によって表される要素でスクリプト本体を実行してから、この変数をリスト内の次の要素に設定します。

以下の **foreach** コマンドの例は、 DCE セルでホストを管理するスクリプトの部分であるとします。このスクリプトの断片は、セル内にホストを構成しているときに障害が起きると、レジストリーからそのホスト・プリンシパル名を除去します。 **foreach** コマンドは、セル内の各プリンシパル名を調べます。 **string** コマンドが **principal catalog** からの出力にリストされているホスト名を検出すると、スクリプトは、レジストリーからプリンシパル名を削除します。

```
foreach princ [principal catalog -simplename] {
    if {[string match $host_name [string range $princ 0 ¥
        [expr [string length $host_name] - 1]]] == 1} {
        principal delete $princ
    }
}
```

ループが結果を戻す先は、インタープリターであり、 **stdout** ではないことに注意してください。そのため、結果を **stdout** へ送るための余分なステップが必要となります。次の例では、**puts** コマンドを使用して、 **foreach** ループの結果を **stdout** に送っています。

```
foreach i [group list subsys/dce/dts-servers] {
    puts [principal show $i]
}
```

また、すべての結果をスクリプト内の 1 つの変数に **append** するか、あるいは **lappend** を使用して、結果を別々のリスト要素として追加することもできます。以下に例を示します。

```
foreach i [group list subsys/dce/dts-servers] {
  append result [principal show $i]
}
return $result
```

while ループ

while ループは、 C の **while** ループと同じように動作します。これは、2 つの引き数、すなわち、式とスクリプト (*body (本体)*) をとります。 **while** コマンドは、本体を実行してから式を再評価し、式が 0 と評価されるまでループを続行します。**while** ループの構文は、次のとおりです。

```
while expression body
```

次の例のプロシージャーでは、 **while** ループを使用して、リスト内の各要素を探索してパターンを見つけます。リスト・サイズに 1 以上の要素が入っていれば (**$size > 0**)、プロシージャーはループを続行します。

```
proc _dcp_list_find {search_list pattern} {

  set found_items ""
  set size [llength $search_list]

  while { $size > 0 } {
    set size [expr $size - 1]
    set index [lsearch $search_list $pattern]
    if { $index == -1 } {
      return $found_items
    }
    lappend found_items [lindex $search_list $index]
    set search_list [lreplace $search_list $index $index]
  }
}
```

for ループ

for ループも、C の for ループと同じように動作します。 **for** は、 **while** よりも複雑ですが、**for** ではループ制御情報がすべて保持されているため、何が起きているかがより分かりやすくなっています。 **for** コマンドの構文は、次のとおりです。

```
for initial_expression test reinit script_body
```

for を使用する場合には、最初の式を設定し、その条件をテストしたあとで、スクリプト本体を実行するようにしてください。スクリプト本体を実行した後、 **for** コマンドは、最初の式を再初期化し、新しい値について再度テストして、テストが偽になるまでループを繰り返します。

次の例は、指定された回数だけオペレーションを行った後、止める **for** ループを示しています。この例では、レジストリーに 50 個のゲスト・プリンシパル名を作成します。

```
dcecp> for {set i 0} {$i < 50}
{incr i} {
> principal create guest$i
> }
dcecp>
```

continue と break によるループの終了

continue および **break** コマンドは、**while**、**for**、および **foreach** コマンドで開始されたループを終了させます。

現行のループ反復を終了させるには、**continue** コマンドを使用してください。たとえば、ループによって、リスト内の特定の要素についてテストし、ある要素については無視して操作を行わず、残りの要素については操作を続行することができます。ループの実行を即時に終了させるためには、 **break** コマンドを使用してください。

次のスクリプトの断片の例は、 **continue** と **break** コマンドを含んだ、**foreach** コマンド・ループです。 **foreach** コマンドは、タイム・プロバイダーの DTS サーバーを検出するまで、セル内のすべての DTS サーバーを調べます。 (タイム・プロバイダーは、外部時刻ソースから時刻を受信する、特殊な DTS サーバーです。) (**dts catalog** オペレーションが作成した) リスト内の最初のサーバーが、 **dts show** オペレーションからの出力を戻した場合は、 **continue** コマンドはスクリプト内の次の行を呼び出します。このスクリプト行は、その出力から **{provider yes}** 属性と値を探索します。 (**attrlist getval** で調べた結果) **provider** 属性が **yes** であれば、スクリプトは **server** 変数をその DTS サブ名に設定し、 **break** コマンドがこの **foreach** ループ全体を終了させます。

```
foreach s [dts catalog] {
    if {[catch {dts show $s} dts_sh_out] != 0} {
         continue
    }
    set p [attrlist getval $dts_sh_out -type provider]
    if {[string match $p "yes"] == 1} {
        set provider "yes"
        set server $s
        break
    }
set provider "no"
}
```

case 実行前のパターンによるテスト

コマンドの中には、ディレクトリー内のオブジェクトのリストやホスト・システムで動作するサーバーのリストといった、リストを戻すコマンドがあります。 **case** コマンドを使用すると、特定パターン (特定オブジェクトまたはサーバーの名前など) がリストまたはストリングにないかどうか、テストすることができます。指定されたパターンを検出すると、 **case** コマンドは、検出されたパターンに関連したスクリプトを実行します。 **case** コマンドの構文は、次のとおりです。

case *string* **in** *pattern* {*script*} *pattern* {*script*}

case コマンドは、 *string* に *pattern* が入っていないか調べて、 {*script*} を実行します。単語の **in** は、省略してもかまいません。次の例は、**case** コマンドの働きを示しています。

```
dcecp> set x {one ten twenty}
one ten twenty
dcecp> foreach el $x {case $el in one {puts script1} two {puts script2}}
script1
dcecp>
```

case コマンドは、まず **$x** に *pattern* **one** が入っていないかを調べます。このパターンが見つかると、関連のスクリプトが **script 1** をディスプレイ上にエコーします。それ以上の一致がなければ、 **case** コマンドは終了します。

より実際的な例として、特定システムのすべてのサーバーをリストする **dcecp** コマンドを実行するとします。このリストを探索して特定のサーバー名を見つけ、それぞれの名前を特定のファイルに追加するスクリプトを実行することができます。

```
case $x in server1 {lappend filename1} server2 {lappend filename2}
```

パターンのリストが長過ぎて数行に渡る場合は、ターゲットのパターンとスクリプトのリスト全体を中括弧で囲むことによって、改行が区切り記号として解釈されないようにすることができます。これには、中括弧で囲まれたリスト内で変数とコマンドの置換を防止できるという利点もあります。

パターンにはワイルドカード文字を組み込むことができます。探索パターン内の ? (疑問符) は、ターゲット・パターン内の任意の 1 文字と一致します。たとえば、**?at** は、 **bat** にも **hat** にも一致します。探索パターン内の * (アスタリスク) は、ターゲット・パターン内の任意のストリングと一致します。たとえば、***at** は、**bat** にも ″**three cornered hat**″ にも一致します (スペースを区切り記号として機能させないようにするには、二重引用符を使用することに注意してください)。

一致するパターンが見つからない場合、デフォルトのスクリプトを実行させる方法が必要な場合があります。 **case** コマンドには、 *default* と呼ばれる特殊なパターンがあります。これの対応するスクリプトは、一致するパターンが見つからない場合に実行されます。デフォルトのパターンは、リスト内の最後に指定してください。

```
case $x in {
  a {puts "script for case a"}
  b {puts "script for case b"}
  default {puts "run this script if no matches are found"}
}
```

コマンドの動的な作成

eval コマンドを使用すると、小さなスクリプトを連鎖させながら、スクリプトを作成することができます。この技法は、いろいろな質問に対する管理者の応答を記録し、それらの応答に基づいた特殊なスクリプトを構成するスクリプトに便利です。構文は次のとおりです。

eval *arg ...arg*

次の例は、変数を使用して、 **account create** オペレーション用のオプションとその値を保持しています。 **eval** コマンドは、変数が拡張し、正常に実行することを保証します。

```
dcecp> set
mpwd {-mypwd mxyzptlk}
-mypwd mxyzptlk
dcecp> set pwd {-password change.me}
-password change.me
dcecp> set org {-organization guests}
-organization guests
dcecp> set grp {-group guest}
-group guest
dcecp> eval account create guest1 $mpwd $pwd $org $grp
dcecp>
```

変数を使用して **eval** コマンドを作成する場合には注意してください。以下のような **eval** コマンドは、スクリプト内に問題を起こすことがあります。原因は、**dcecp** が解析を 2 度行うためです。最初に、**dcecp** は、 **eval** コマンドとその引き数を解析します。次に、それらがスクリプトとして実行されるときに **eval** 引き数を再度解析します。

```
dcecp> eval $a $b $c
dcecp>
```

次の例のように、引き数の前後に中括弧を指定することによって、解析の問題を回避できることがあります。

```
dcecp> eval {$a $b $c}
dcecp>
```

dcecp に **eval** コマンドを正しく解析させるためには、 **dcecp list** コマンドを起動して、次のような有効なリスト構造を生成します。

```
dcecp> eval [list $a $b $c]
dcecp>
```

dcecp スクリプトとしての他のファイルの読み取り

source コマンドは、他のファイルの内容を読み取り、それらを **dcecp** スクリプトとして実行します。この機能を使用すると、構築ブロックのように、低レベルの機能をつなぐことによって、高レベルの機能を構成することができます。スクリプトを修正して複製するのではなく、再利用するため、スクリプトの開発と保守は容易になり、しかも一貫したものになります。このコマンドの構文は次のとおりです。

source *filename*

source からの戻り値は、 *filename* の最後のコマンドからの戻り値です。

実際的な例を挙げると、あるスクリプトでは CDS サブツリー内の項目をリストし、別のスクリプトではサブツリーを削除し、さらにもう 1 つのスクリプトではサブツリーを移動するとします。これらのすべてのスクリプトが必要とする共通の機能は、サブツリーのルートの下にすべての子ディレクトリーをリストすることです。すべての子をリストするスクリプトを作成して、それに **children_list.dcp** のような名前を与えることができます。 (**.dcp** 拡張子は、スクリプト・ファイルに命名する場合の **dcecp** 規則です。) スクリプトのいずれかがすべての子ディレクトリーをリストする必要がある場合は、次のように単に **source** コマンドを使用します。

```
source children_list.dcp
```

return コマンドを使用して **source** コマンドを終了させます。 **return** コマンドは、予期したエラーまたは予期しないエラーが発生した場合にも、 **source** や **proc** などのコマンドが制御された方法で終了できる方法を提供します。 **return** コマンドは、エラー状態によりスクリプト全体を終了させたり、失敗させたりするのではなく、エラー情報を管理して、スクリプトが実行を継続できるようにします。この他のエラー処理技法で **return** を使用する方法については、 46ページの『エラーおよび例外の処理』で説明します。

新しいコマンドの作成

DCE 制御プログラムには、 DCE オペレーションを制御しモニターするための強力で広範囲なコマンドのセットが用意されています。しかし、エンド・ユーザーによる DCE の厳密な使用法については、予測できません。したがって、非常に特殊な要件に合わせるために、管理者が追加のコマンドを作成しなければならない場合があります。 **proc** コマンドは、 **set** 、**list**、および **while** のような組み込みコマンドに外観も動作も似た、コマンドを追加作成するための簡単な方法を提供します。しかし、**proc** を使用して作成するコマンドは、 C で書かれた組み込みコマンドとは異なり、次に示すようにスクリプトを使用して書かれます。

```
dcecp> proc div {x y} {expr $x/$y}
dcecp>
```

proc は、プロシージャー名、プロシージャー引き数の名前のリスト、および新しいプロシージャーの本体を形成する **dcecp** スクリプト、の 3 つの引き数を使用します。次の新しいプロシージャー **div** には、2 つの引き数が必要です。たとえば、次のとおりです。

```
dcecp> div 12 4
3
dcecp>
```

デフォルトにより、 **proc** はすべての変数がローカル変数であると想定します。すなわち、変数の名前と値は、プロシージャー内でのみ設定され、プロシージャーの完了時にそれらは期限切れになります。次のコマンドは、プロシージャー内で変数 *x* と *y* が設定されていないためにエラーとなります。

```
dcecp> set x 15
15
dcecp> set y 3
3
dcecp> proc div {} {expr $x/$y}
dcecp> div
Error: cannot read "x": no such variable
```

global コマンドを使用すると、グローバル変数 (プロシージャーの外側で定義された変数) をインポートすることができます。

```
dcecp> set
x 15
dcecp> set y 3
dcecp> proc div {} {
> global x y
> expr $x/$y
> }
dcecp> div
5
dcecp>
```

グローバル変数は、インポート後、そのプロシージャーの期間中、存続します。プロシージャーは、**unset** と **set** を使用して、変数の値を変更することができます。新しい値は、プロシージャーの内側および外側で、使用することができます。次の例を参照してください。

return コマンドを使用すると、プロシージャーを直ちに戻すことができます。**return** に対する引き数の値が、プロシージャーの戻り値になります。

```
proc find {a} {
 <some pattern matching script that looks for a specific CDS entry>
    if {a != b} {
       return 1
    }
    return 0
}
```

引き数を 1 つも取らないプロシージャー、あるいは可変の数の引き数を取るプロシージャーを設計することもできます。たとえば、引き数を取らないプロシージャーは、次の例のように、単純なオペレーションをいくつか実行するだけです。

```
proc _do_create_group {} {
     global rpcgroupname
     rpcgroup create $rpcgroupname
}
```

引き数リスト内でネストしたリスト構造を使用することによって、引き数のデフォルト値を指定することもできます。次の例では、最初の引き数 *attr* は必ず指定しなければなりません。 2 番目の引き数 *value* は、指定しなければ、**unset** がデフォルトになります。

```
proc _attr_show {attr {value "unset"}}  {
     puts "$attr is $value"
}
```

プロシージャーは他のプロシージャーをコールすることができます。現行プロシージャーは、 **upvar** コマンドを使用することによって、任意の呼び出し側プロシージャーから変数をインポートすることができます。例を示します。

```
upvar level otherVar1 myVar1 otherVar2 myVar2
```

level 引き数が **1** ならば、親プロシージャーの変数コンテキストが入手されます。この引き数が **2** ならば、親の親のプロシージャーの変数コンテキストが入手されます。 *level* 引き数の前に **#** を置くことによって、グローバル・コンテキストとの相対的な関係でレベルを指定することもできます。 *level* が **#0** の場合は、グローバル変数が入手されます。 *level* が **#1** の場合は、グローバル・レベルから起動されたプロシージャーからの変数が入手されます。

otherVar 引き数は、インポートしたい変数の名前を指定します。現行プロシージャーで使用するために変数の名前を変更するには、 **myVar** 引き数を組み込む必要があります。次の例は、インポートした変数を **cargs** という名前に変更します。

```
upvar 1 local_args cargs
```

プロシージャーは、**uplevel** コマンドを使用して、親プロシージャーのコンテキストのもとでスクリプトを実行させることもできます。このコマンドは、プロシージャーのコンテキストを管理するための便利な方法を提供します。たとえば、親プロ

シージャーから多くの変数をインポートして操作するのではなく、 **uplevel** を使用すれば、それらの変数に一度に接続することができます。構文は次のとおりです。

uplevel *level arg arg arg*

uplevel コマンドは、**eval** と同様に、引き数を連結してスクリプトとして実行します。しかし、**uplevel** は **eval** と異なり、現行コンテキストではなく、 *level* によって指定されたコンテキストでスクリプトを実行します。 *level* 引き数は、 **uplevel** 内で、**upvar** 内の場合と同様に、機能します。親のコンテキストを使用するには、**1** の *level* 引き数を使用します。第 1 レベルのプロシージャーのコンテキストを使用するには、 **#1** の *level* 引き数を使用します。

すでに有効であるコマンド名を **proc** コマンドで指定すると、新しいプロシージャーが、同じ名前の既存のプロシージャーに置き換わります。特別な場合を除き、既存の組み込みコマンドを置き換えるような新しいコマンドの命名はしないでください。

rename コマンドを使用すると、 Tcl コマンドの名前の変更または削除を行うことができます。たとえば、 **list** を一時的に **list.old** に変更してから、**proc** を使用して、 **list** という名前の別のコマンドを作成することができます。作成した **list** コマンドの使用が完了した後、 **list.old** を **list** に変更して、 **list** の元の機能を復元することができます。以下に例を示します。

```
rename list list.old
proc list {} {
     <some list operation>
}
rename list.old list
```

rename コマンドに対する 2 番目の引き数を省略すると、コマンドが削除されます。次の例は、**list** コマンドを削除します。

```
rename list
```

ストリング処理

多くの DCE 管理オペレーションは何らかの情報を戻します。たとえば、 **principal show** オペレーションは、プリンシパルについての情報を戻します。次の例のように、通常、この情報はリスト形式です。

```
dcecp> principal show R_Parsons
{fullname {}}
{uid 15}
{uuid 0000000f-d6f9-21cd-8d00-0000c08adf56}
{alias no}
{quota unlimited}
{groups users}
dcecp>
```

管理者がリストをスキャンしてそのリストから必要な情報を取り出すのは非常に簡単ですが、スクリプトの場合は方法が異なります。スクリプトは、特定の情報を探索する場合、通常、リストの概念を無視し、リストを構成する文字の集合 (**ストリング**と呼ばれる) に働きかけます。 DCE 制御プログラムにはストリングに対して動作するコマンドのセットが備えられているため、ストリングの構成、解析、比較、値の取り出し、および変更を行うことができます。

ストリングの構成

多くの場合、スクリプトは、その他のコマンドで使用するため、あるいはユーザーのスクリーン上に表示するために、ストリングを構成する必要があります。 DCE 制御プログラムは、スクリプトで使用するためのストリングを構成する場合に使用される、 **format** コマンドを提供します。

format コマンドは、必要に応じて変数を置換します。次の例では、 **format** コマンドを使用して **_dcp_host_entries** 変数を構成し、セル名のストリング (**string (ストリング)** のタイプは **%s** によって示されます) をストリング **/hosts** の前に付けています。セル名は、**c** コンビニエンス変数に含まれています。

```
dcecp> set _dcp_host_entries [format "%s/hosts" $_c]
/.../my_cell.goodco.com/hosts
dcecp>
```

format コマンドは、10 進数、8 進数、16 進数、浮動小数点、および浮動小数表記を含む、いろいろな形式相互間で引き数を変換することもできます。また、符号付き数値の符号の印刷または削除、出力の右寄せまたは左寄せ、およびスペースまたはゼロの埋め込みを指定することもできます。次の例では、整数 8 がそれに相当する 8 進数の数値に変換されます。 2 番目の例では、出力を右へ 9 文字スペース分だけシフトします。

```
dcecp> format
%1o 8
10
dcecp> format %9o 8
       10
dcecp>
```

ストリングの解析

DCE 制御プログラムには、ストリングを解析してから変換し、ストリングの関係する部分を変数に保管する **scan** コマンドが含まれています。この機能は、たとえば、前のコマンドによって戻された情報を、別のコマンドへ入力することができるデータに変換する場合に便利です。 **scan** コマンドの構文は、次のとおりです。

scan "*string*" "*format*" [*varname* [*varname*]...]

string はリテラルで指定することも、あるいは変数を使用して指定することもできます。 *format* のセクションは、構文解析を制御します。このとき、読みやすくするために *format* セクションに入れられたブランクやタブ文字は無視されます。このセクションは、 % (パーセント記号) で区切られた、1 つまたは複数の変換指定子で構成されています。変換指定子は、変換のタイプを指定するとともに、ストリングのどの部分を変換し保管するかも定義します。

次の例では、 **_dcp_temp** に含まれているストリングを有効な浮動小数点数に構文解析し、それを変数 **_dcp_temp2** に保管します。

```
if { [scan $_dcp_temp "%f" _dcp_temp2] != 1 } {
   error "Variable ¥"$_dcp_temp¥" is not a ¥
     valid floating-point number"
}
```

その他のストリング・オペレーション

string index および **string range** を使用すると、ストリング内の 1 文字または ある範囲の文字を指定することができます。これらのコマンドは、予測可能な長さ のストリングから情報を抽出する場合に便利です。

string index コマンドには、ストリングから抽出される 1 つの文字の位置 (ゼロから始まり、左から右へカウントしたもの) を示す、 1 つの引き数を指定します。 **string range** コマンドには、その範囲に含める左端の文字および右端の文字の位置を表す、 2 つの引き数を指定します。次の例は、**string range** コマンドの 1 つの使用法を示します。

```
dcecp> string range {The
quick brown fox} 4 9
quick
dcecp>
```

string compare を使用すると、あるストリングが、辞書編集上 (アルファベット順で) 別のストリングより大きいか、小さいか、あるいは等しいかを判別することができます。通常、このオペレーションでは、ストリングを構成する ASCII コードのバイト比較を行います。

string length コマンドを使用すると、ストリング内の文字数をカウントすることができます。例を挙げます。

```
dcecp> string length "The
quick brown fox"
19
dcecp>
```

string toupper と **string tolower** コマンドを使用すると、文字を大文字、小文字に変換できます。例を挙げます。

```
dcecp> string toupper "The quick brown fox"
THE QUICK BROWN FOX
dcecp>
```

string trim コマンドを使用すると、ストリングから特定の文字をトリムする (落とす) ことができます。また、**string trimleft** および **string trimright** コマンドを使用すると、ストリングの左端または右端の文字 (複数も可) を削除することができます。

パターン・マッチングは、いくつかの方法のうちのいずれで行うこともできます。″glob″ スタイルのパターン・マッチングは、 **string match** コマンドで呼び出すことができます。これは、 **csh** で利用可能な glob パターン・マッチング機能に似たものであり、一致すれば **1**、一致しなければ **0** を戻します。 **regexp** コマンドを使用すると、さらに柔軟性が高い正規表現のパターン・マッチング (**egrep** で見られるような) を行うことができます。 **regsub** コマンドを使用して、このオペレーションを拡張し、正規表現の置換を行うことができます。

次の例は、**regsub** コマンドの使用法を示すものです。最初の引き数は探索パターンを指定します。 2 番目の引き数は、探索するストリングです。 3 番目の引き数は置換パターンを指定します。最後の引き数は、 **regsub** が新しいストリングを置く、変数です。このコマンドは、置換が行われなかった場合は **0** を戻し、置換が行われた場合は **1** を戻します。

```
dcecp> regsub brown
"The quick brown fox" blue color
1
dcecp> puts $color
The quick blue fox
```

エラーおよび例外の処理

dcecp のインタープリターは、 **dcecp** スクリプトで問題が生じたときにエラー情報を戻す、エラー機能を備えています。エラー情報は、今後同じ間違いをしないように、問題の内容をユーザーに知らせます。 **dcecp** エラーの原因は、いろいろです。たとえば、コマンドが正しい数の引き数を受け取らなかったり、コマンドに何らかの入力ミスがあったり、あるいはオペレーションのオブジェクト (CDS ディレクトリーなど) が何らかの理由で利用不能である場合などです。

ここでは、エラーおよび例外を処理するための、次の 3 つの方法について説明します。

- グローバル・エラー情報変数の使用
- 例外の catch
- 複合エラーの再現

グローバル・エラー情報変数の使用

dcecp はエラーを検出すると、次のような、説明メッセージを印刷します。

```
Error: wrong # args: should be "set varName ?newValue?"
```

エラー・メッセージだけでは、問題が発生した個所を正確に判別できないことがあります。そこで、**dcecp** は、 **errorInfo** という名前のグローバル変数に追加のエラー情報を保管します。ユーザーのスクリプトでこの情報をアクセスし印刷して、エラーの検出に役立てることができます。一般的には、これは、エラーの発生時に実行中であったコマンドをトレースしたものです。

次の例は、 **errorInfo** に保管できる情報の種類を示しています。これを逆の順序で読み取ることによって、 **user** オペレーションの **_dcp_create_user** プロシージャーから呼び出された、 **parseargs** プロシージャー内のスクリプト本体の 4 行目近くでエラーが発生したと判断することができます。

```
dcecp> puts $errorInfo
Unknown option "group"
    while executing
"
    invoked from within
"
    ("while" body line 4)
    invoked from within
"
    (procedure "parseargs" line 60)
    invoked from within
"
    (procedure "_dcp_create_user" line 64)
    invoked from within
"
    invoked from within
"
    invoked from within
```

```
"
    (procedure "user" line 24)
"
dcecp>
```

さらに、**dcecp** は、 **errorCode** という名前の別のグローバル変数に、別の種類のエラー情報を保管することもできます。この変数には、他のクラスのエラーを識別できる、次に示すようなリストが入っています。

```
UNIX, ENOENT, "insufficient arguments for filename"
```

DCE 制御プログラムは、エラーが有益なエラー情報を生成しない場合は、**errorCode** 変数を NONE に設定します。

エラーと例外をトラップするための catch の使用

場合によっては、ある種のエラーの場合は活動状態のコマンドを終了させないで、これらのエラーをトラップすることができます。 **catch** コマンドを使用すると、エラーをトラップし、無視することができるので、スクリプトに処理を続行させることができます。例として、コマンドが存在すれば、スクリプトでそのコマンドの名前を変更したい場合を考えます。しかし、**rename** コマンドを実行するときに、コマンド名が存在しないこともあります。

```
dcecp> rename move move.old
Error: cannot rename "move":  command does not exist
dcecp>
```

catch を使用して、 **rename** コマンドをスクリプトとして呼び出します。

```
dcecp> catch {ren move move.old}
1
dcecp>
```

catch コマンドは、その引き数をスクリプトとして扱って、それを実行し、正常に実行されると **0** を戻します。エラーが起きた場合は、 **catch** コマンドがエラーをキャッチして、**1** を戻します。

catch コマンドには、 2 番目の引き数を追加することができます。この引き数は、スクリプトの戻り値 (正常終了時) またはエラー・メッセージを保持するために **catch** が変更する、変数です。 **catch** コマンドの構文は、次のとおりです。

catch *command varName*

スクリプトでの **catch** の用途の 1 つは、他のプロシージャーを呼び出すことです。下記のスクリプトの断片は、次のように読むことができます。
「**_dcp_create_group** プロシージャーが失敗した (**!= 0**) 場合は、**_dcp_cleanup_user_create** プロシージャーを実行して、 **msg** 変数に保管されているエラーを表示する。」

```
if {[ catch {_dcp_create_group $group group_created} msg] != 0 } {
      _dcp_cleanup_user_create $element -principal
          error $msg
}
```

例外は、**break**、 **continue** および **return** コマンドが生成する、特殊クラスのエラーです。 **break** および **continue** コマンドは、**while**、 **for** および **foreach** などのループを終了するために使用し、また、**return** コマンドは、 **proc** または **source** コマンドを終了するために使用します。

ループが、より包括的なコマンドの内部で (その一部として) 存在するようなプロシージャーでは、結果として起きる例外が処理しにくい場合があります。たとえば、CDS 内の特定のオブジェクト・タイプを探索するユーザー作成プロシージャーが、特定の属性のオカレンスについてテストするために、ループ活動の一部として **foreach** を呼び出すことがあります。

break、**continue** または **return** コマンドを使用してループの実行を管理したり、他のネストされたコマンド (たとえば、 **case** や **if**) を管理したりする場合、親コマンドは例外をキャッチできる状態にはなりません。親コマンドは打ち切って、通常どおりにエラー・メッセージを出します。ただし、エラーは親コマンドに関連しており、そのエラーが実際に起きたループ・コマンドまで追跡することは困難です。

他のコマンドによってコールされたコマンドを終了させるために、 **continue**、**break** または **return** コマンドを使用する必要があるときは、 **catch** を使用してネストされたコマンドを呼び出し、ここから **continue**、**break**、または **return** コマンドを呼び出してエラーまたは例外からの回復を図る方法をとってください。**catch** コマンドをこのように使用すると、例外を、追跡がより簡単な、ループまたはネストされたプロシージャー内にとどめることができます。

```
foreach s [server catalog] {
    if {[catch {server show $s} srv_sh_out] != 0} {
        continue
    }
```

複合エラーの再現

proc コマンドを使用すると、非常に正確なオペレーションを実行するプロシージャーまたはコマンドを作成することができます。たとえば、サーバーの実行に関する情報を検索し、フィルターにかける、 **_dcp_get_servers** と呼ばれるユーザー作成プロシージャーには、特定のストリングを求めてサーバー情報を探索するループなどの、いろいろなサブタスクを実行する、ネスト されたコマンドまたはプロシージャーを入れることができます。ネストされたコマンドまたはプロシージャーを使用すると、包括的なプロシージャーまたはコマンドを開発することができますが、エラーが正しく知らされない場合、特定するのが難しいエラーを生成する場合があります。

複合スクリプトで **error** コマンドを使用すると、すでに実行されたスクリプトの一部によって生成されたエラーを再現することができます。次のスクリプトの断片は、ハード・コーディングされたエラー・メッセージを単に印刷するものです。このような使用によって、エラー状態を正確に記述する、特製のメッセージを作ることもできます。

```
set dts_cat_out [_dcp_dts_catalog]
if {[llength $dts_cat_out] == 0} {
    error "Unable to find any DTS servers"
}
```

次のスクリプトの断片は、さらに多くのことを行います。すなわち、**catch** を使用して、 **_dcp_create_group** プロシージャーから戻されたエラー情報を **msg** 変数に保管します。障害が発生する (**!= 0**) と、スクリプトは、実行された部分を元に戻すクリーンアップ・プロシージャーを呼び出してから、 **msg** 変数に保管されているメッセージを印刷します。

```
if {[ catch {_dcp_create_group $group group_created} msg] != 0 } {
      _dcp_cleanup_user_create $element -principal
      error $msg
}
```

これまで、非常に単純なエラー処理技法について説明してきました。しかし、特に複合度の高い状態では、エラー処理も複雑になることに注意してください。エラー処理の詳細については、 Tcl のより一般的な使用について記載している、他の資料をお読みになることをお勧めします。

ファイルの処理

DCE 制御プログラムには、ファイルからの読み取りと書き込みに使用されるコマンドがいくつもあります。ファイルは、後で参照するために **dcecp** オペレーションの出力を保管するような場合に便利です。次に、ファイル操作についていくつかの有用な例を示します。

- セル内のすべてのホストにわたって **server catalog** オペレーションを実行し、ホスト固有のファイルに各ホストからの結果を保管することができます。後で、ファイルを比較して、サーバー構成に関するレポートを作成することができます。
- 各アカウントが最後にログインされた時刻を示す **dcecp** スクリプトを実行し、後で評価するためにファイルにこの情報を保管することによって、非活動状態のアカウントを検出することができます。
- **dcecp hostdata** オブジェクトを使用することにより、簡単には操作できない DCE ファイルを変更することもできます。たとえば、新しい属性を **cds_attributes** ファイルに追加する機能を作成することができます。

OSF によって提供される DCE は、現在、 UNIX スタイルのシステム、または POSIX システム・コールをサポートするシステムのためのファイル・オペレーションのみをサポートします。ただし、一部のベンダーの DCE バージョンには、他のシステム上でのファイル・オペレーションをサポートするものがあります。

ファイル名の指定

通例の UNIX 規則を使用してファイル名を指定してください。たとえば、**/opt/dcelocal/dcecp/server_snap.dcecp** は、 **/opt/dcelocal/dcecp** というディレクトリーにある **server_snap.dcecp** という名前のファイルを指します。また、**~dce_admin/scripts/server_snap.dcecp** および **~/admin/server_snap.dcecp** のような相対ファイル名を使って、ファイルを参照することもできます。 **pwd** コマンドを使用すると、現行作業ディレクトリーを印刷することができ、 **cd** コマンドを使用すると、現行作業ディレクトリーを設定することができます。次のコマンドは、現行ディレクトリーを **~dce_admin/scripts** と設定します。

```
dcecp> cd ~dce_admin/scripts
dcecp>
```

glob コマンドを使用すれば、ディレクトリー内のファイルのリストを表示することができます。このコマンドは、パターン引き数に一致するファイル名のリストをコマンドに戻します。例を挙げます。

```
dcecp> glob *
help local_lib.dcp
dcecp>
```

file コマンドにいろいろなオプションを指定して使用すると、ファイルに関するその他のいろいろな情報を表示することができます。 **file** コマンドは、ファイルの通過パス数、サイズ、または許可 (ファイルが現行ユーザーによって実行可能か、読み取り可能か、書き込み可能かどうか) に基づいてファイルを選択するのに役立ちます。

ファイルの読み取りと書き込み

ファイルの読み取りと書き込みのための **dcecp** コマンドは、 C 言語の対応する **fopen**、**fclose** などと、構文も動作も似ています。

open コマンドを使用すると、読み取りおよび書き込みのためにファイルをオープンすることができます。 **open** コマンドの 2 番目の引き数 (以下の例では **+r** となっている) は、ファイル・アクセス・モードを指定します。読み取りまたは書き込み、もしくはその両方のためにファイルをオープンすることができ、既存のファイルを置き換えるか、新しい情報を指定して既存のファイルに追加するかを指定することができます。また、最初のアクセス位置をファイルの先頭かファイルの終わりに設定することもできます。デフォルトのアクセス・モードは、読み取り専用 (ファイルはすでに存在している必要があります) です。

```
dcecp> open server_snap.dcecp
+r
file5
dcecp>
```

open コマンドは、ファイルのオープン時に各ファイルにファイル識別子を割り当てます。後続のコマンドでファイルを参照する場合は、このファイル識別子を使用してください。

ファイルのオープン後、**puts** コマンドを使用すると、ファイルに行を追加することができます。通常は、**dcecp** は、十分なデータを蓄積するまで待機してから、この情報をファイルに書き込みます。 **dcecp** が直ちに情報をファイルに書き込むようにしたい場合には、 **flush** コマンドを使用します。ファイルから次の行を読み取るには、 **gets** を使用し、ファイルから多数のバイトまたはすべてのバイトを読み取るには、 **read** を使用します。次の例では、すべてのプリンシパルのリストを **prins** という名前のファイルに書き込みます。

```
dcecp> open prins w+
file8
dcecp> puts file8 [principal catalog]
dcecp> close file8
dcecp>
```

場合によっては、ファイルの最初の行から読み取りや書き込みを開始したくない場合があります。 DCE 制御プログラムには、アクセス位置を設定するためのコマンドがいくつか用意されており、したがって、ファイル内のすべての行を通過する必要はありません。これらのコマンドをランダム・アクセスをサポートしない端末装置やその他の順次装置のような装置で使用すると、エラーになります。ファイル内のアクセス・ポイントを設定するには、 **seek** コマンドを使用します。起点からのバイト数でオフセットを指定してください。起点は、ファイルの先頭または終わ

り、または現在位置とすることができます。ファイルの先頭方向に移動するには、負の数を使用します。この例では、現行のアクセス位置から 16 バイト戻ります。

```
dcecp> seek file5
-16 current
dcecp>
```

tell コマンドを使用すると、現行のアクセス位置を判別することができます。後でファイル内のその位置に戻ることができるように、変数に戻り値を保管してください。

最後に、次のように **close** コマンドを使用すると、ファイルをクローズすることができます。

```
dcecp> close file5
dcecp>
```

サブプロセスの生成

サブプロセスを使用してコマンドを実行すると、複合スクリプト記述または特殊な管理上のニーズに対するいくつかの便利な解決法が得られます。サブプロセスでは次のものを提供することができます。

- オペレーティング・システム・コマンドへのアクセス
- 正常な同期実行を確立する方法
- 複合スクリプトまたは複雑なスクリプトの効率を上げる方法

スクリプトからのオペレーティング・システム・コマンドの実行

DCE 制御プログラムは多様性に富んでいますが、スクリプトでオペレーティング・システム・コマンドを使用して、簡単な (またはそれほど簡単ではない) オペレーションを実行したい場合があります。 **exec** コマンドは、コマンドを実行するサブプロセスを作成することによって、外部コマンドを実行するための方法をスクリプトに提供します。次の例は、 **exec** コマンドを使用して、ローカル・ホスト名を検索します。このホスト名は *hostname* 変数として設定され、以後、スクリプトで使用されます。

```
dcecp> set hostname
[exec hostname]
myhost
dcecp> directory list /.:/hosts/$hostname -simple
cds-clerk cds-server dts-entity profile self
dcecp>
```

exec コマンドは、通常、サブプロセスで実行されたオペレーションの結果を戻します。しかし、 UNIX のリダイレクト記号 (<、<<、>) を使用して、標準入力または標準出力をリダイレクトすることができます。また、| (縦線) 記号を使用すれば、**nroff**、 **sort**、または **grep** などのフィルターを通して、出力をパイプ処理することができます。

exec コマンドは、単独で使用するときは同期的です。すなわち、外部コマンドが完了してから、スクリプトは実行を継続します。しかし、CDS セル内のディレクトリーを同期化するときのように、サブプロセスが完了するまでに長い時間がかかる場合は、 **exec** コマンドを & 記号とともに使用して、サブプロセスをバックグラウンドに押しやることができます。次の例は、**exec** コマンドを使用して、以前に収集

した出力をプリンターに送信します。この場合、スクリプトは、**print** コマンドが完了するのを待たずに、実行を継続します。

```
dcecp> exec lpr output.log &
dcecp>
```

第3章 スクリプトおよび dcecp オブジェクトの作成

DCE 制御プログラムには、 DCE セル内の管理可能な各コンポーネントへの管理アクセスを提供する、多くのオブジェクト が用意されています。たとえば、**principal** オブジェクトによって、管理者は、セキュリティー・サービス・レジストリー・データベース内のプリンシパル情報を管理することができます。同様に、**rpcgroup** オブジェクトによって、管理者は、CDS 内のグループ情報を管理することが可能となります。

いくつかの DCE オペレーションは、新しいユーザーを DCE セルに追加するために複数のオペレーションを実行する必要がある場合のように、複数のコンポーネントに影響を与えます。この要件を満たすために、DCE 制御プログラムには、管理者が 1 つのオペレーションで複数のコンポーネントを処理するためのタスク・オブジェクト が用意されています。たとえば、**user** タスク・オブジェクトは、レジストリー内でのプリンシパル情報の作成、組織および関連グループへのプリンシパルの追加、ユーザー用の CDS ディレクトリーの作成など、いくつかのオペレーションを実行します。タスク・オブジェクトは、他の **dcecp** オブジェクトと外観および動作が同じであり、他の **dcecp** オブジェクトと同じヘルプ・システムを使用します。しかし、タスク・オブジェクトは、 C プログラミング言語の代わりに、**dcecp** 言語を使用して作成されます。このため、管理者は、既存のスクリプトの拡張またはカスタマイズを容易に行えます。

DCE 制御プログラムには、いくつかの複数コンポーネントのオペレーションを扱うためのタスク・オブジェクトが用意されていますが、セル構成の相違および管理者がセルを管理する方法の相違のために、用意されている DCE タスク・オブジェクトがすべての DCE セルのすべての要件を満たすことは実際には困難です。たとえば、いくつかのセルは DFS コンポーネントを使用することがあり、また、あるセルは標準の OSF DCE のセル・ディレクトリー・ネーミング構造とは異なる構造を使用することがあります。また、DCE のインストール先によっては、管理のために個別の **dcecp** オブジェクトを必要とする、サービスやリポジトリーなどの特別な管理コンポーネントを持っている場合があります。

管理者は、DCE 制御プログラム言語を使用すれば、セル固有の要件を満たすために、独自のスクリプトを作成することができます。管理者は、既存のタスク・オブジェクトの拡張または変更、あるいは、 DCE セル内の特別なコンポーネントを管理するための新しいタスク・オブジェクトの作成も行うことができます。この章では、次に示す種類の dcecp スクリプトの拡張、変更、または作成について説明します。

- 非公式管理スクリプト
- 公式タスク・オブジェクト

非公式管理スクリプト

管理者は、非公式管理スクリプトによって、複数のオペレーションをファイルに格納し、必要に応じて再利用することが可能となります。非公式スクリプトは、 1 つか 2 つの引き数しか取らないオペレーション、または単純なタスクしか実行しないオペレーションに便利です。さらに、非公式スクリプトの正確な動作および出力

は、作成者の要件に合わせてカスタマイズし、調整することができます。非公式スクリプトは、セル内の管理者間で共用できますが、通常は、作成者の **.dcecprc** ファイルの中に組み込まれています。

スクリプトは、一般的に、**proc** コマンドによって作成された、 1 つまたは複数のプロシージャーからなります。これによって、**dcecp** プロンプトにプロシージャーの名前を入力するだけで、スクリプト化されたオペレーションを起動することができます。

次に示す簡単なスクリプトは、現行セルおよびログイン識別に関する情報を印刷します。

```
# Show your current login name and your current cell name.
proc _dcp_whoami {} {
  global _c _u
  puts stdout "You are '$_u' logged into '$_c'."
}
```

このスクリプトは、直接、**.dcecprc** ファイルに組み込むか、または **source** コマンドを使用し、実際のスクリプトを外部ファイルに保持することによって、**.dcecprc** ファイルに組み込むことができます。 2 番目の方法を使用すれば、他の管理者も同じスクリプトを **source** コマンドを使用して指示するだけで、自分の **.dcecprc** ファイルに組み込むことができます。この方法では、**.dcecprc** ファイルが整然と保持されるので、何が行われているかを他の人が容易に理解できます。別の方法として、スクリプトまたはポインターを **init.dcecp** ファイルに置くことができます。このファイルに対する変更は、ホスト上のすべてのユーザーが使用できます。 **init.dcecp** ファイルおよび **.dcecprc** ファイルの詳細については、 3ページの『第1章 DCE 制御プログラムの紹介』の 18ページの『dcecp セッションのカスタマイズ』を参照してください。以下は、**.dcecprc** ファイルの **source** コマンドの例です。

```
source /usr/users/wardr/dcecp/local_lib.dcp
```

.dcp ファイル名拡張子は、 DCE 制御プログラムによって使用されるファイルのネーミングのための規則です。もう 1 つの規則では、**_dcp_whoami** のように、プロシージャー名の前に **_dcp** を付けます。多くの **dcecp** プロシージャーは、この規則を使用する必要のないユーザー作成のプロシージャーと区別するために、この規則に従っています。 **_dcp_whoami** のような、覚えたり入力したりするのが難しいプロシージャー名の場合は、名前を変更することができます。たとえば、このプロシージャーの名前を、以下のように **.dcecprc** ファイルで **rename** コマンドを使用して、 **whoami** に変更することができます。

```
rename _dcp_whoami whoami
```

変更を取り込むためには、**dcecp** プログラムを再起動してください。これで、以下のような DCE 制御プログラムのプロンプトに、 **whoami** と入力することができます。

```
dcecp> whoami
You are 'cell_admin' logged into '/.../my_cell.goodco.com'.
dcecp>
```

オペレーションを連鎖することによって、さらに多くのことを実行するスクリプトを作成することができます。たとえば、以下のスクリプトは、DCE セル内のすべてのホストをリストします。次に、このスクリプトは、dts エンティティーに関する

CDS の中に、各ホストがオブジェクト・エントリーを持っているかどうかをチェックします。 (これは、DTS サーバーがそのホストで使用可能であることを示します。) DTS エンティティーに関するオブジェクト・エントリーが存在するホストごとに、このスクリプトは、当該ホスト上の時刻を戻す、**clock show** オペレーションを実行します。このスクリプトは、読みやすいようにフォーマットして画面上に情報を出力し、セル内のすべてのホストについてすべてのホスト・エントリーのチェックが済むまでループし続けます。

前に説明した簡単なスクリプトと同じようにして、 **_dcp_show_clocks** プロシージャーを **dcecp** セッションの中で使用できるようにしてください。

```
# Show the time on all of the dts servers running in your cell.
proc _dcp_show_clocks {} {
    set x [directory list /.:/hosts]
    foreach n $x {
        if {[catch {object show $n/dts-entity}] == 0} {
            set index [string last "/" $n]
            set y [string range $n [incr index] end]
            if {[catch {clock show $n/dts-entity} msg] == 0} {
                set i [expr 20 - [string length $y]]
                puts [format "Time on $y is %${i}s %s" " " ¥
                    [clock show $n/dts-entity]]
            } else {
                set i [expr 20 - [string length $y]]
                puts [format "Time on $y is %${i}s %s" " " ¥
                    "Server not responding."]
            }
        }
    }
}
```

公式タスク・オブジェクト

DCE 環境によっては、標準の DCE 制御プログラム・オブジェクトでは厳密に対処できない特別な管理要件があるものがあります。このような管理要件に合う非公式スクリプトを作成して分散することができますが、これらのオペレーションを何らかの方法で文書化する必要があるかもしれません。しかし、さらに重要なことは、複雑なオペレーションの場合、スクリプトの動作を正確に制御するには多くのオプションを使用する必要があることです。ヘルプ情報を提供し、オペレーションを解析する複雑な引き数を処理するための、独自のメカニズムを作成するのではなく、既存のヘルプ・システムおよび **dcecp** で提供される他の公式タスク・オブジェクトが利用する **parseargs** 機能を使用することができます。この方法によれば、他の **dcecp** オブジェクトとの整合性があるスクリプトを作成することができます。

公式タスク・オブジェクトは、上記の非公式スクリプトの概念に、以下のいくつかの重要な事項を追加して構築されます。

- スクリプトの先頭にある引き数テーブルは、スクリプト内の個別のプロシージャーとしてオペレーションを定義します。引き数テーブルは、使用可能なオプションも定義します。 **parseargs** プロシージャーが、スクリプトの呼び出し時にスクリプトに渡される引き数とオプションを解析するために、呼び出されます。
- 各オペレーションに関するヘルプ情報が、スクリプト内の引き数テーブルに置かれます。他のスクリプト・ユーザーは、標準の **dcecp help** オペレーションを使用して、この情報を入手することができます。

- スクリプトが実行される条件を予想し制御することはできないので、広範なエラー制御が組み込まれています。

この項の残りの部分では、公式タスク・オブジェクト内で使用される一般的な構造と規則を示します。説明のために、 DCE 制御プログラムとともに提供される **dcecp user** タスク・オブジェクトを使用します。

タスク・オブジェクトのモデル

この項では、DCE 制御プログラムとともに使用するために作成され、他のタスク・オブジェクト内でエミュレートされる **user** タスク・オブジェクトの部分について調べます。基本モデルに従うことによって、タスク・オブジェクトの外観と動作は、 **dcecp** の他の部分との一貫性を保てます。

効率よく、また読みやすくするために、例には **user** タスク・オブジェクトに含まれているすべてのプロシージャーを示してはいません。さらに、組み込みプロシージャーのいくつかの反復部分を省略して、コード例の中ではその省略部分は垂直に並んだ省略符号でもって置き換えられています。 **user** タスク・オブジェクト全体は、*dcelocal*/**dcecp** に入っています。

オブジェクトの名前は、″show″ または ″modify″ などの動詞ではなく、操作対象のエンティティーに基づいて付けてください。 DCE 制御プログラム・オブジェクトの名前は、操作対象の DCE エンティティーに基づいて付けられています。 **rpcentry** および **principal** のようなプリミティブ・オブジェクトは、管理可能な単一の DCE エンティティーを操作します。タスク・オブジェクトは、一般的に、目的を達成するために複数のプリミティブ・オブジェクトを呼び出して、高位で操作します。 user タスク・オブジェクトの作成者は、管理可能なオブジェクトとして、より高位のエンティティー (*user*) を考案しました。

user オブジェクトは、 **user** オブジェクトが提供するプロシージャーとオペレーションを定義する、最高位の **proc** コマンドとその引き数テーブルで始まっています。以下の構文を使用して、この引き数テーブル内で個別のプロシージャーを定義してください。

```
verb command function_call  procedure_name "helptext_string"
```

parseargs プロシージャー (**parseargs.dcp** と呼ばれる別個のファイルに定義されている) への呼び出しは、その引き数とともに呼び出される内部プロシージャーの名前を戻します。 **parseargs** プロシージャーについては、 61ページの『parseargs プロシージャーの使用』に説明があります。

```
# proc user - This procedure is the front end for the user task
# scripts.  All argument checking for the provided switches is done
# in the individual functions.
#
proc user { args } {
  set arg_table {
    {create command function_call _dcp_create_user
       "Create a DCE user" }
    {delete command function_call _dcp_delete_user
       "Delete a DCE user"}
    {show   command function_call _dcp_show_user
       "Show the attributes of a DCE user"}
    {help   help  help_list
       "Print summary of command-line options and abort"}
    {operations operations operation_list
```

```
       "Return valid operations for command."}}
  set verbose_prose
"This object allows the manipulation of a DCE user.  A user is
represented as a principal and account with membership in a group and
organization as well as having a directory in the CDS namespace.  A user
may be created, deleted or have attribute information returned.  The
argument is a list of either relative or fully qualified principal names.
All fixed attributes of the principal and account object may be specified
when creating a user.  The -force option to the create verb allows the
group or organization for that user to be created if necessary.  The user
is provided a directory in the CDS namespace, with the appropriate ACLs.
Access to create a user requires the correct ACLs on principal, group and
organization directories within the registry and the clearinghouse and
users directory in the CDS namespace."
  set local_args $args
  parseargs $arg_table local_args -found_one

  if { [info local help_prose ] > 0 } { return $help_prose }
  if { [info local function_call ] > 0 } {
    return [$function_call local_args]
  } else {
    error "¥"user¥" object requires a verb to form a command."
  }
}
```

スクリプトの次の部分は、入力として多くのオプションまたは属性を取る、**_dcp_create_user** というプロシージャーを調べます。このプロシージャーは、ユーザーを作成する実際の作業を行うために多くの下位プロシージャーを使用しますが、この例では、その下位プロシージャーの 1 つである **_dcp_create_principal_entry** だけを示すことから開始しています。

その後に、スクリプトには **_dcp_create_user** プロシージャーが続いています。このプロシージャー (およびすべての下位プロシージャー) の名前は、下線から始まっていることに注意してください。これが、Tcl **info** コマンドが頻繁に使用されて、すべてのプロシージャーの名前を戻している理由です。この規則によって、このような内部プロシージャー名と、文書化されるプロシージャーである **user** のようなプロシージャーが区別されます。さらに、名前の **_dcp** 部分によって、**dcecp** プロシージャーとホスト上の他の Tcl プロシージャーが区別されます。

_dcp_create_user プロシージャーには、有効なオプションを定義する引き数テーブルがあります。この引き数テーブルがスクリプトの初期引き数テーブルと異なる点は、 **command** キーワードおよび、スクリプト内の個別のプロシージャーを定義する **function_call** 変数がないことです。

次に、スクリプトは、リスト内のオプションまたは属性として入力された変数を初期設定します。 **process_attribute_list** プロシージャー (例の最後にあります) は、リストとして渡された属性を実際に解析します。次に、スクリプトは、レジストリー内と CDS 内にユーザー情報を作成する作業を行います。クリーンアップ・プロシージャー **_dcp_cleanup_user_create** (終わりの近くにあります) で、失敗したユーザー作成オペレーションをやり直すことができます。

```
 .
 . [several low-level procedures omitted]
 .
#
# This procedure creates a principal in the current registry _s(sec)
# if that principal does not yet exist.
#
proc _dcp_create_principal_entry { principal_name princ_args} {
```

```
  set list_of_principals [principal catalog]
  if { [lsearch $list_of_principals $principal_name] == -1} {
    if { [llength $princ_args ] != 0 } {
      principal create $principal_name -attribute $princ_args
    } else
  } else {
    error "Principal ¥"$principal_name¥" already exists."
  }
}
#
# proc _dcp_create_user - This procedure actually creates a DCE user.
# Several steps are performed.  If the principal does not exist
# a new one is created.  If the groups do not exist and a -force switch is
# set, then two new groups will be added.  The user will be added to the
# groups.  The account will then be created.  An entry in the CDS
# namespace will then be created with the appropriate ACLs.
#
proc _dcp_create_user { local_args } {
  set arg_table {
    {-alias string alias
          "Add principal named as an alias of specified uid."}
    {-attribute string attribute_list
          "Provide attributes in an attribute list format."}
    {-client string client
          "Can the account principal be a client."}
    {-description string descr
           "A general description of the account."}
    {-dupkey string dupkey
           "Can the accounts' principal have duplicate keys."}
    {-expdate string expdate
           "When does the account expire."}
 .
 . [repetitive elements omitted]
 .
    {-uid integer uid
         "User Identifier of the principal to be added."}}
#
# Initializing some variables.
#
    upvar 1 local_args cargs
    set local_args $cargs
    set account_args ""
    set princ_args ""
    set group_args ""
    set force 0
    parseargs $arg_table local_args -no_leftovers
    if { [info local help_prose ] > 0 } { return }
    if { [llength $local_args] > 1 } {
      error "Unrecognized argument [lindex $local_args 1]."
    } elseif { [llength $local_args] == 0 } { error "No user name."
    } else { set account_name $local_args }
#
#  If parseargs returned attributes in a list instead of options,
#  create an attribute list.  Then call process_attribute_list to
#  parse the list.
#
    if { [info local attribute_list] > 0} {
       set pile_of_attributes "alias client descr dupkey expdate¥
       forwadabletkt fullname force group home organization maxtktlife ¥
       maxtktrenew mypwd password postdatedtkt proxiabletkt pwdvalid ¥
       renewabletkt server quota shell stdgtauth"
       process_attribute_list attribute_list $pile_of_attributes
    }
#
# If user entered attributes as options rather than in a list,
# check for attribute options.
#
```

```
    if { [info local group] > 0} {
      set account_args [format "%s {%s %s}" $account_args group $group]
    } else { error "No group name specified." }
    if { [info local organization] > 0} {
      set account_args [format "%s {%s %s}" $account_args organiz ¥
  $organization]
    } else { error "No organization name specified." }
    if { [info local password] > 0} {
      set account_args [format "%s {%s %s}" $account_args password ¥
  $password]
    } else { error "No password specified." }
    if { [info local mypwd] > 0 } {
      set account_args [format "%s {%s %s}" $account_args mypwd $mypwd]
    } else { error "No admin password specified." }
#
# principal and group operations both use the principal's fullname
#
    if { [info local fullname] > 0 } {
      set princ_args [format "%s {%s {%s}}" $princ_args fullname ¥
  $fullname]
      set group_args [format "%s {%s {%s}}" $group_args fullname ¥
  $fullname]
    }

    if { [info local uid] > 0 } {
      set princ_args [format "%s {%s %s}" $princ_args uid $uid]
    }
 .
 . [repetitive elements omitted]
 .

    if { [info local stdtgtauth] > 0 } {
       set account_args [format "%s {%s %s}" $account_args stdtgtauth ¥
           $stdtgtauth]
    }
#
# set variables if entered as attributes in an attribute list
#
    set account_name [lindex $account_name 0]
    set group_created 0
    set org_created 0
    set group_arg ""
    set org_arg ""
#
# do the work - create principal, do group and organization
# operations, create the account, and create directory in CDS
#
    foreach element $account_name {
      set clup_user "_dcp_cleanup_user_create $element -principal"
      _dcp_create_principal_entry $element $princ_args

      if { $force == 1 } {
   if {[ catch {_dcp_create_group $group group_created} ¥
    msg] != 0 } {
              _dcp_cleanup_user_create $element -principal
              error $msg
          }
          if { $group_created == 1 } {
             set group_arg "-group group"
          }
          if {[ catch {_dcp_create_org $organization org_created} ¥
    msg] != 0 } {
              set clup_user [concat $clup_user $group_arg]
              eval $clup_user
              error $msg
          }
          if { $org_created == 1 } {
```

```
            set org_arg "-org organization"
        }
    }
    set clup_user [concat $clup_user $group_arg $org_arg]
    if {[catch {_dcp_add_group_entry $group  $element} msg] != 0} {
        eval $clup_user
        error $msg
    }

    if {[catch {_dcp_add_org_entry $organization $element} msg] != 0 }
{
  eval $clup_user
        error $msg
    }

    if {[catch {_dcp_add_account_entry $element $account_args} ¥
msg] != 0} {
        eval $clup_user
        error $msg
    }

    if {[catch {_dcp_add_namespace_entry $element} msg] != 0} {
        eval $clup_user
        error $msg
    }
   }
   set _n $account_name
   return
}
#
# _dcp_cleanup_user_create  - This function undoes changes after a
# failure in one of the user create functions as though the operation
# never occurred
#
proc _dcp_cleanup_user_create {account_name args} {

    if { [lsearch $args -principal] != -1 } {
       principal delete $account_name
    }
    if { [lsearch $args -group] != -1 } {
       upvar 1 group clean_group
       group delete $clean_group
    }
    if { [lsearch $args -org] != -1 } {
       upvar 1 organization clean_org
       organization delete $clean_org
    }
}
#
# process_attribute_list - Takes an attribute_list and parses out the
#                          appropriate attributes contained in the
#                          pile_of_attributes variable
#
proc process_attribute_list {attribute_list pile_of_attributes} {
   foreach element $pile_of_attributes { upvar 1 $element _dcp_$element
}
   upvar 1 attribute_list _dcp_attribute_list
   set _dcp_attribute_list [check_list_list $_dcp_attribute_list]
   foreach element $_dcp_attribute_list {
       if { [llength $element] != 2 } {
           error "Incorrect attribute list element
       set attribute_name [lindex $element 0]
       set attribute_value [lindex $element 1]
       set _dcp_attr_name [info vars _dcp_$attribute_name*]
       if {[llength $_dcp_attr_name] > 1} {
           error
            "Ambiguous attribute "¥$attribute_name¥" could be:
```

```
$_dcp_attr_name."
       }
       set [set _dcp_attr_name] $attribute_value
   }
}
proc check_list_list {attribute_list} {
   set not_list_list 0
   set i 1
   foreach element $attribute_list {
        if {[llength $element] != 2 && [llength $attribute_list]
< 3} {
           if {$i == 1} {
              return [format "{%s}" $attribute_list]
           }
        }
        incr i
   }
   return $attribute_list
}
```

次に説明する **user** タスク・オブジェクト内のプロシージャーは、 1 つのオプションの引き数を受け取って、多くの出力情報を戻す **_dcp_show_user** プロシージャーです。このプロシージャーは、 **principal show** および **account show** の各オペレーションの結果を戻します。

```
#_dcp_show_user - This procedure shows the principal and account
#                 attribute lists for a specified user.
#
proc _dcp_show_user {local_args} {
    upvar 1 local_args cargs
    set local_args $cargs

    parseargs "" local_args -no_leftovers
    if { [info local help_prose ] > 0 } { return }
    if { [llength $local_args] > 1 } {
      error "Unrecognized argument [lindex $local_args 1]."
    } elseif { [llength $local_args] == 0 } { error "No user name."
    } else { set account_name $local_args }
    # Take the first element of the account_name in order to
    # eliminate list nesting.
    set account_name [lindex $account_name 0]
    set _dcp_principals [principal catalog -simplename]
    # Show each account that has been requested.
    foreach element $account_name {
      if { [lsearch $_dcp_principals $element] == -1 } {
error "User ¥"$element¥" does not exist."
      } else {
set _dcp_user_attributes [principal show $element]
      }
      set _dcp_accounts [account catalog -simplename]
      if { [lsearch $_dcp_accounts $element] == -1 } {
error "User ¥"$element¥" does not exist."
      } else {
set _dcp_user_attributes [format "%s¥n%s" ¥
    $_dcp_user_attributes [account show $element -all]]
      }
    }
    return $_dcp_user_attributes
}
```

parseargs プロシージャーの使用

引き数またはオプションを受け取るタスク・オブジェクトおよびスクリプトは、オブジェクトまたはスクリプトの呼び出しとともに渡される引き数を解析するため

に、 **parseargs** プロシージャーを呼び出すことができます。 **parseargs** プロシージャーは、 **dcecp** スクリプト内の引き数解析のために便利で再使用可能な方法を提供する、個別ファイル内のスクリプトです。基本構文は次のとおりです。

```
parseargs parse_options local_args args
```

このプロシージャーは、呼び出し側スクリプトによって渡された引き数を使用します。 **parseargs** プロシージャーには、以下の入力が必要です。

parse_options
解析オプションについて記述する引き数テーブル (**arg_table**)。 *parse_options* 引き数は、スクリプトの最高位の引き数テーブル内と同様に、5 つの要素から、あるいはスクリプト内の呼び出されるプロシージャーの下位引き数テーブル内と同様に、 4 つの要素から、構成することができます。 *parse_options* に関する 2 つの構文は、以下のとおりです。

```
verb command variable command_name "help string"
```

または

```
-options type variable "help string"
```

verb 最高位の解析を提供します。一般に、1 つのオペレーションには 1 つのオブジェクトと 1 つの動詞が含まれます。動詞部分は、一般に、別のプロシージャーを呼び出します。

command
定義中のプロシージャーが、オブジェクトの動詞であることを示すキーワード。

variable
オプションの値を保持する変数の名前。動詞を解析する場合、変数は**関数呼び出し**の名前になります。オプションを解析する場合、変数の名前は、解析するオプションの名前になります。たとえば、オプション名が **-alias** であれば、変数は、**alias** という名前になります。

command_name
変数に格納するプロシージャー名。

help string
動詞またはオプションの使用法について記述するストリング。

-options
-attribute または **-mypwd** など、解析するオプションの実際のストリングの値。

type **-option** に関連づけられる変数のタイプ。使用可能なタイプは、**integer**、**string**、**float**、**boolean**、**command**、および **help** です。

local_args
解析する引き数。 **parseargs** プロシージャーは、認識されたすべてのエントリーを抽出してリストにし、解析されなかった (または解析できなかった) 値を使用して *local_args* をリセットします。たとえば、**user create** のよ

うな最高位のコマンドには、後から、**create** オペレーションを実行するプロシージャーがスクリプト内で呼び出された時に解析されるオプションが含まれています。

args 以下の 2 つのフラグのうちの 1 つ。

-found_one
プロシージャー引き数が 1 個検出されたら戻るように解析プログラムに指示します。たとえば、**user create** では、構文解析プログラムは、**create** コマンドが 1 つ検出され処理された後に戻ります。

-no_leftovers
余分なオプションがないか探索し、オプションを見つけた場合、エラーを生成します。

タスク・オブジェクトの起動

タスク・オブジェクトが作成 (およびテスト) されたら、それを使用できるようにする必要があります。スクリプトを個人的に使用する場合は、個人の **.dcecprc** ファイルに入れて、 53ページの『非公式管理スクリプト』で説明したようにして起動してください。

公式タスク・オブジェクトの場合は、他の **dcecp** オブジェクトと同様に、動作させるためのいくつかのステップが必要です。

1. **root** としてログインし、完成したスクリプトを *dcelocal***/dcecp** ディレクトリーにコピーし、ファイル許可を実行可能に設定します。
2. **dcecp** を開始し、**auto_mkindex** ユーティリティーを実行します。これによって、使用可能なすべてのオブジェクトについて DCE 制御プログラムに通知する情報が作成されます。 root 権限を使用して、タスク・オブジェクトが常駐しているディレクトリー内で次のコマンドを実行してください。 UNIX システムでは、多くの場合、そのディレクトリーは *dcelocal***/dcecp** です。

   ```
   % dcecp
   dcecp> auto_mkindex /opt/dcelocal/dcecp *.dcp
   dcecp>
   ```

3. 新しいタスク・オブジェクト名を **dcecp** のヘルプ画面に組み込むには、ファイル **/opt/dcelocal/dcecp/help.dcp** を編集してください。このファイルは、**dcecp help** オペレーションに応答して表示されます。

このファイルは、スクリプトを実行する各 DCE ホストで使用できるようにする必要があります。一般に、これは、各ホストの **/opt/dcelocal/dcecp** ディレクトリーへのファイルのコピーと、このディレクトリー内の各ファイルに対する **auto_mkindex** ユーティリティーの実行を意味します。**/opt/dcelocal/dcecp/help.dcp** ファイルにオブジェクト名も入れたい場合があります。

便利な方法として、各ホスト上にファイルを作成するために、 DCE 制御プログラムの **hostdata** オブジェクトを使用するスクリプトを書くことができます。こうすれば、このスクリプトでは、 **hostdata** オブジェクトのポストプロセッサー属性を使用して、 **auto_mkindex** ユーティリティーを実行することができます。 103ページの『第9章 DCE ホスト・サービスとホスト・データの管理』に、 **dcecp hostdata** オブジェクトの使用法についての説明があります。

第2部 DCE 管理タスク

第4章 DCE 管理タスクのオブジェクト

IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド ― コア・コンポーネント のこの部分では、 DCE と一緒に提供される DCE 管理タスク・オブジェクトの目的と使用法について説明します。一般的に、これらの特別な **dcecp** オブジェクトは、複数の下位のオペレーションを結合することによって、日常的な高位の管理タスクを実行します。

1 つのタスク・オブジェクトが、複数の DCE サービスを使用したり、これらのサービスに影響を与えることは頻繁にあります。たとえば、タスク・オブジェクトの 1 つである **host** オブジェクトは、ホスト・コンピューターを DCE セルの中に構成することができます。このタスクは、特定の種類の情報を DCE セキュリティー・サービス、セル・ディレクトリー・サービス、および DCE ホスト・デーモン・サービスに追加します。 **host** オブジェクトを 1 回起動するだけで複数のステップを実行することができるので、 DCE 管理者は、複数の下位の **dcecp** 管理オブジェクトを使用して介入しなければならないところを、下位の管理上の詳細に立ち入らなくて済みます。

高いレベルのタスク・オブジェクトについて説明しているときには、詳細は説明していませんが、それ以上のことが行われている場合が多いことに注意してください。このような場合、このガイドで詳細情報を参照できる個所を示します。通常は、本書の関連するコンポーネントの部分の対応する下位の説明を参照することになります。

DCE 管理を単純化するためのタスク・オブジェクトの使用

DCE 制御プログラムの各オブジェクトは、DCE 内の情報の特定の部分を操作します。たとえば、**group** オブジェクトは、セキュリティー・サービス・レジストリー・データベース内のセキュリティー・グループだけを操作します。 **group** オブジェクトは、セキュリティー・グループの作成と削除、セキュリティー・グループへのメンバーの追加と削除、グループの名前変更などを、管理者が行えるようにします。このような厳密な制御は、特定のニーズまたは環境に合わせて DCE をカスタマイズするために不可欠です。

このような制御は、新しいセルを構成する場合、またはアクセス制御問題を解決する場合などに必要ですが、これにより日常的な DCE 管理タスクを処理することもできます。 1 例として、新しいユーザーを DCE セルに追加するために必要な、最低限のステップを次に示します。

1. **principal** オブジェクトを使用して、ユーザーに関するプリンシパル名を作成します。
2. **group** オブジェクトを使用して、プリンシパルをセキュリティー・グループに追加します。
3. **organization** オブジェクトを使用して、プリンシパルをセキュリティー組織に追加します。
4. **account** オブジェクトを使用して、プリンシパルに関するアカウントを作成します。

5. **directory** オブジェクトを使用して、プリンシパル用のディレクトリーを CDS 内に作成します。
6. **acl** オブジェクトを使用して、 CDS ディレクトリーへのアクセス権をプリンシパルに与えます。

上記の 6 つのステップを実行する場合、15 人から 20 人のユーザーがいる小さなセルでは、特に問題はありません。しかし、百人または千人あるいはそれ以上のユーザーがいるセルの場合には、上記のステップおよび他の管理タスクの自動化が必要になります。

この管理要件を満たすために、DCE 制御プログラムには、日常的な DCE 管理タスクを実行するための、いくつかの管理タスク・オブジェクト が組み込まれています。ここでは、6 つの下位オペレーションの実行からなるユーザーの追加の場合のように、複数のステップを必要とするものを実行していることを意味するために、タスク という用語を使用しています。

タスク・オブジェクトのうちの 1 つは、 DCE 環境内のユーザー情報を追加および削除するために使用することができる、 **user** オブジェクトです。たとえば、**user** オブジェクトを 1 回起動するだけで、新しいユーザーを DCE 環境に正しく追加するために必要な、上記の 6 つのステップ全部を実行することができます。この同じタスク・オブジェクトを使用して、環境からユーザーを削除することもできます。

タスク・オブジェクトは、 DCE 制御プログラム言語を使用することによって、**dcecp** スクリプトとして実装されます。つまり、ユーザー固有の要件に合わせてスクリプトを拡張したり、その動作を変更できることを意味します。たとえば、**user** タスク・オブジェクトのデフォルトの実装は、 DFS のどの情報も操作しません。このような拡張サービスが DCE 環境に含まれている場合、 DFS の一部のオペレーションをスクリプトに追加したい場合があります。このガイドの 1ページの『第1部 DCE 制御プログラム』で、DCE 制御プログラム言語を使用して **dcecp** タスク・オブジェクトの作成と変更を行う方法について説明しています。

ツールを超えた視点

タスク・オブジェクトを使用して各種の管理オペレーションを実行できますが、最も重要な焦点は、管理対象の要素またはエンティティーです。 DCE に用意されている 4 つのタスク・オブジェクトのそれぞれによって、 DCE セル内の特定の要素またはエンティティーを管理することができます。各要素を次に示します。

DCE セル
: **cell** タスク・オブジェクトを使用すれば、セルが動作中かどうかをテストし、セル内で利用可能なサービスに関する一般情報を表示し、セキュリティー情報および CDS 情報をバックアップすることができます。

セル名 セルの別名を作成し管理することができます。この別名は、複数のグローバル・ディレクトリー・サービス内にセルを登録するのに必要です。これらのオペレーションには、**cellalias** タスク・オブジェクトを使用します。

DCE ホスト
: **host** タスク・オブジェクトを使用すれば、セル内の DCE ホストを構成および削除し、セル内のホストに関する情報を表示し、セル内のホスト上の DCE プロセスを開始および停止することができます。

DCE ユーザー

user タスク・オブジェクトを使用すれば、ユーザーを追加および削除し、DCE セル内のユーザーに関する情報を表示することができます。

この部の残りの章では、 DCE とともに提供される 4 つの **dcecp** タスク・オブジェクトのデフォルトの実装によって、これらの DCE 要素を管理する方法について説明します。

第5章 DCE セルの管理

セル管理者の視点から見ると、DCE セルは、分散アプリケーションの実行をサポートする、 1 組のネットワーク化されたサービスからなります。しかし、このような単純な言葉では、セル内で現在どのようなサービスが使用可能であるかについて、実際には何も言い表すことはできません。実際に、DCE サーバーの正確な個数およびその位置はセルによって異なります。同じセルであっても、ホストおよびネットワークの故障および再構成によって、サービスの可用性は異なります。

さまざまなサービス関連 **dcecp** オブジェクトを使用すれば、セル内でサービスが使用可能かどうか、また、それがどこで使用可能かをテストすることができますが、これはかなり面倒な作業になります。その代わりとして、DCE 制御プログラムには、構成済みの DCE サーバーをリストし、サービスが使用可能であるかどうかをテストする、**cell** タスク・オブジェクトが用意されています。このタスク・オブジェクトは、DCE セキュリティー・サービスおよび CDS によって保守される、重要なデータのバックアップも行うことができます。

構成済みのすべての DCE サーバーと DCE ホストの表示

追加または削除される DCE ホストまたは DCE サーバーが少なく、比較的安定している DCE セルがあります。逆に、ホストおよび DCE サーバーが毎週または毎日、追加、削除、または移動されて、非常に動的なセルもあります。このような環境では、 DCE リソースの場所を追跡することが困難な可能性があります。このため、**cell** タスク・オブジェクトには **show** オペレーションがあり、セル内のさまざまなデータベースをスキャンして、構成済みの DCE サーバーおよび DCE ホストの名前を戻します。

cell show コマンドの 1 つの用途は、パフォーマンス問題の追跡です。たとえば、多くの新しいホストとユーザーが追加されているが、 CDS またはセキュリティー・サーバーの数や位置がそれに対応して増加されていない場合があります。あるいは、新しいセルの管理担当者になったばかりで、セルのコンポーネントを知りたい場合があります。

セル内に構成されている DCE サーバーおよびホストを表示するには、 **cell show** オペレーションを入力します。このコマンドは、次に示すような、タイプ別にまとめたサーバーのリストと、 DCE ホストのリストを戻します。

secservers
: 各値はセキュリティー・サーバーの名前です。

cdsservers
: 各値は、CDS サーバーを実行中のマシンの名前です。この名前は、***/.:/hosts*** の下にある単純名です。クリアリングハウスは、このマシン上で構成する必要があります。

dtsservers
: 各値は、セル内の DTS サーバーの名前です。

hosts 各値は、上でサーバーとして示されたマシンを含む、セル内のホストの名前です。これは、単に **directory list /.:/hosts** オペレーションの戻り値です。

次の例は、ローカル・セル内に構成されている、すべての DCE サーバーおよびホストの名前を示します。

```
dcecp> cell show
{secservers
 /.../my_cell.goodco.com/subsys/dce/sec/earth}
{cdsservers
 /.../my_cell.goodco.com/hosts/earth}
{dtsservers
 /.../my_cell.goodco.com/hosts/krypton}
{hosts
 /.../my_cell.goodco.com/hosts/earth
 /.../my_cell.goodco.com/hosts/jupiter
 /.../my_cell.goodco.com/hosts/krypton
 /.../my_cell.goodco.com/hosts/mars
 /.../my_cell.goodco.com/hosts/mercury
 /.../my_cell.goodco.com/hosts/neptune
 /.../my_cell.goodco.com/hosts/pluto
 /.../my_cell.goodco.com/hosts/saturn
 /.../my_cell.goodco.com/hosts/uranus
 /.../my_cell.goodco.com/hosts/venus}
dcecp>
```

必要な許可を持っていれば、次の例に示したように、引き数として別のセルの名前を組み込むことによって、そのセル内に構成されている DCE サーバーおよびホストを表示することができます。

```
dcecp> cell show
/.../their_cell.goodco.com
{secservers
 /.../their_cell.goodco.com/subsys/dce/sec/gold}
{cdsserver
 /.../their_cell.goodco.com/gold}
{dtsservers
 /.../their_cell.goodco.com/hosts/silver/dts-entity}
{hosts
 /.../their_cell.goodco.com/hosts/brass
 /.../their_cell.goodco.com/hosts/bronze
 /.../their_cell.goodco.com/hosts/copper
 /.../their_cell.goodco.com/hosts/gold
 /.../their_cell.goodco.com/hosts/iron
 /.../their_cell.goodco.com/hosts/mercury
 /.../their_cell.goodco.com/hosts/silver
 /.../their_cell.goodco.com/hosts/steel
 /.../their_cell.goodco.com/hosts/tin}
dcecp>
```

セル・オペレーションのテスト

クライアント / サーバーの通信問題が発生した場合、セル内で 1 つまたは複数の DCE サービスが作動していないと推測するのが普通です。 **cell ping** オペレーションを起動することによって、セルの DCE サービスが実行されているかどうかを簡単にテストすることができます。

オプションを指定せずに呼び出した場合、**cell ping** オペレーションは、マスター・セキュリティー・サーバー上、マスター・クリアリングハウスがある CDS サーバー上、およびセル内のすべての DTS サーバー上で、**server ping** オペレーション

を実行します。 CDS およびセキュリティー・サービスのマスターだけでなくレプリカもテストするには、 **-replicas** オプションを使用します。 **-clients** オプションを使用した場合、 CDS の **/.:/hosts** ディレクトリー内でループを行い、各ホスト名を引き数として使用して **host ping** を実行することによって、セル内のすべての DCE ホストをテストすることができます。

このオペレーションが失敗した場合、エラーが生成され、テストできなかったサーバーまたはホストのリストが戻されます。成功した場合は、オペレーションは DCE Services Available (DCE サービス使用可能) というメッセージを戻します。 **-clients** オプションを使用して成功した場合は、メッセージは、DCE Clients Available (DCE クライアント使用可能) になります。

次の例は、ローカル・セル内に構成されている、すべてのマスター DCE サーバーの名前を ping します。

```
dcecp> cell ping
DCE services available
dcecp>
```

次の例は、ローカル・セル内に構成されているすべての DCE ホストの名前を ping します。セルのサイズおよび設定されているタイムアウト値によって、このコマンドは完了するまでに長い時間 (数分から数十分) かかる場合があります。

```
dcecp> cell ping -clients
DCE clients available
dcecp>
```

必要な許可を持っていれば、次の例に示したように、引き数として別のセルの名前を組み込むことによって、そのセル内に構成されている DCE サーバーおよびホストを ping することができます。

```
dcecp> cell ping /.../their_cell.goodco.com
DCE services available
dcecp>
```

セキュリティー・サービス・レジストリーと CDS のバックアップ

注: 以下の説明は、既存の DCE のみに適用されます。 DCE セキュリティー・レジストリーおよび LDAP 統合機能に関しての情報は、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

日常のオペレーションについてオーガニゼーションの DCE セルに対する依存度が高まるにつれて、セルのディレクトリーおよびセキュリティー・データを失う余裕はなくなります。オーガニゼーションは、通常、こうしたデータやその他の重要なデータを失わないように、定期的なバックアップ方式を使用しています。しかし、従来のバックアップ方法を使用してこうした DCE データベースをバックアップすると、アーカイブが適切に保護されていない場合には、セル内にセキュリティー上の欠陥が生じる可能性があります。

幸いにも、DCE には、ユーザーが選択した宛先へこうした重要なデータベースをバックアップする機能が組み込まれています。この DCE の機能を使用して CDS と

セキュリティー・データのバックアップを開始した後、従来のバックアップ・プログラムをリダイレクトして、こうした DCE データベースを無視させることができます。

cell backup オペレーションは、マスター・セキュリティー・データベース、およびセル内のマスター・レプリカを持つ各クリアリングハウスをバックアップします。このオペレーションの場合、バックアップされる各サーバー・ホストで **dced** プログラムが実行中であることが必要です。

バックアップの宛先 (通常、テープ・アーカイブ) を指定することができる拡張レジストリー属性 (ERA) をセットアップすることによって、定期的なバックアップ・オペレーションのためのセルを準備をします。次に、マスター DCE セキュリティー・サービス・レジストリー・データベース、およびバックアップしたいマスター・レプリカをもつすべての CDS クリアリングハウスについてプリンシパルに新しい属性を追加します。これを行うには、以下のステップにしたがってください。

1. セル管理者として、 1 つの ERA をバックアップの宛先を指定するストリングとして作成します。 **/.:/sec/xattrschema/bckp_dest** という名前の ERA とタイプ **printstring** を指定します。 **principal** という名前の ACL マネージャーを選択し、その 4 つの許可ビットを、以下のコマンドに示すように、 **r** (**読み取り**)、**m** (**管理**)、 **r** (**読み取り**)、および **D** (**削除**) に設定します。

   ```
   dcecp> xattrschema create /.:/sec/xattrschema/bckp_dest ¥
   > -encoding printstring -aclmgr {principal r m r D}
   dcecp>
   ```

2. セル管理者として、新しい ERA (**bckp_dest**) をプリンシパル **dce-rgy** (DCE セキュリティー・サービス・レジストリー・データベース) に追加します。値を、**tar** ファイル名またはバックアップの宛先である装置に設定します。

   ```
   dcecp> principal modify dce-rgy -add {bckp_dest tarfilename_or_device}
   dcecp>
   ```

3. セル管理者として、新しい ERA (**bckp_dest**) をプリンシパル **/.:/hosts/***hostname***/cds-server** (CDS サーバー) に追加します。値を、**tar** ファイル名またはバックアップの宛先である装置に設定します。

   ```
   dcecp> principal modify /.:/hosts/hostname/cds-server ¥
   > -add {bckp_dest tarfilename_or_device}
   dcecp>
   ```

CDS サーバーをバックアップし、リストアするためには、 CDS サーバーの中の以下のディレクトリーをバックアップし、リストアします。

```
/opt/dce/local
/var/dce
/krb5
/etc/dce
```

これで、レジストリー・データベースまたは CDS データベースをバックアップしたい場合は、以下のように **cell backup** オペレーションを起動するだけですみます。

```
dcecp> cell backup
dcecp>
```

セル名を **cell backup** オペレーションへの引き数として組み込むことによって、別のセルをバックアップすることができます。リモート・セルでは許可が必要なこと

に注意してください。 (必要とされる特権については、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

DCE サーバーの IP アドレスの変更

サーバーの更新

CDS およびセキュリティー・サーバーとして構成されたマシンの IP アドレスを変更するには、以下の手順を使用してください。

1. DFS サーバーまたは何らかの DCE アプリケーションがシステムで実行されている場合はそれらを停止し、 AIX の場合には **inittab** の中の、また Solaris の場合には **/etc/rc*.d** の中の **auto-restart** を使用不可にします。
2. マシン上のクリアリングハウスの知識を除去します。これは、IP アドレスが変更された後で回復されます。名前がわからない場合には、 **cdscp show cell /.:** コマンドを使用して名前を入手します。

   ```
   cdscp clear clearinghouse /.:/<host_ch>
   ```

3. マシン上のすべての DCE デーモンを停止します。

   ```
   stop.dce
   ```

4. 1 つのコマンドで、エンドポイント・データベース、クラーク・キャッシュ、および古い証明書を次のコマンドで除去します。

   ```
   clean_up.dce
   ```

5. 新しいアドレスを反映させるために **/opt/dcelocal/etc/security/pe_site** ファイルを編集して、セキュリティーを開始できるようにします。
6. **opt/dcelocal/var/dced/cdscache.inf** が存在する場合には、その中の IP アドレスを変更して、 **cdsadv** がディレクトリー・サーバーを見つけられるようにします。
7. システム上の IP アドレスを変更して、リブートします。 inittab でもって DCE を自動的に開始できない場合は、 **start.dce** を使用して開始してください。まだ CDS が完全に機能していないため、 **gdad** および **dtsd** デーモンは立ち上がりません。これらのデーモンは、変換プロセスが完了した後で開始してください。システムが **dced** 登録を検査するとき (待機メッセージが表示されているとき) に、 <**Ctrl-C**> を押して終了してください。
8. CDS が使用可能でないため、**BIND_PE_SITE** 環境変数を設定します。

   ```
   export BIND_PE_SITE=1
   dce_login cell_admin
   ```

9. それが管理するクリアリングハウスを CDS に対して識別します (前の **clear clearinghouse** コマンドで使用したのと同じ名前を使用していることを確認します)。

   ```
   cdscp create clearinghouse /.:/<host_ch>
   ```

10. 開始した時点では CDS サーバーはクリアリングハウスがどれであるかがわからないため、 **cdsadv** プロセスも、このクリアリングハウスが存在することがわかりません。新しいアドレスを反映させるために **/opt/dcelocal/etc/security/pe_site** ファイルを編集して、セキュリティーを開始できるようにします。クラーク・キャッシュを再構築します。

```
stop.dce cds_srv cds_cl cds_second
cd /opt/dcelocal/var/adm/directory/cds
rm cds_cache.* cdsclerk_*
start.dce cds_srv cds_cl cds_second
```

システムが dced 登録を検査するとき (待機メッセージが表示されているとき) に、 <**Ctrl-C**> を押して終了してください。

11. これで、CDS およびセキュリティー・サーバーは、新しい IP アドレスを使用するよう再構成されました。 **BIND_PE_SITE** 環境変数の設定を解除し、**dce_login** が通常どおりできることを検証します。

```
unset BIND_PE_SITE
dce_login cell_admin
```

12. ネームスペースをうまくアクセスできることを確認します。

```
     cdsli -o
/.:/cell-profile
/.:/fs
/.:/lan-profile
/.:/sec
```

13. CDS のサーバー・セルフ (self)・エントリーを更新します。システムも DFS サーバーまたは、そのセルフ・エントリーに依存する他のアプリケーションを実行している場合には、このステップは重要です。

```
rpccp unexport -i elaf8308-5dlf-11c9-91a4-08002b14a0fa,3.0 ¥
               /.:/hosts/<server_name>/self
rpccp export -i elaf8308-5dlf-11c9-91a4-08002b14a0fa,3.0 ¥
                -b ncadg_ip_udp:<new_ip_addr>[135] ¥
               /.:/hosts/<server_name>/self
```

14. **stop.dce** を実行します。
15. **start.dce** を実行します。

クライアントの更新

CDS またはセキュリティー・サーバーのいずれかの IP アドレスが変更された後は、以下の変更をクライアント・マシンで行う必要があります。 DFS または何らかの DCE アプリケーションがシステムで実行されている場合は、以下のステップを実行する前に、それらを停止する必要があります。

1. マシン上のすべての DCE デーモンを停止します。

```
stop.dce
```

2. エンドポイント・データベース、クラーク・キャッシュ (これは CDS サーバーの古い IP アドレスを参照しています)、およびセキュリティー証明書を除去します。

```
clean_up.dce
```

3. **/opt/dcelocal/etc/security/pe_site** ファイルを変更して、再起動時に **dced** がセキュリティー・サーバーを見つけられるようにします。
4. **/opt/dcelocal/var/dced/cdscache.inf** ファイルの中の IP アドレスを変更して、**cdsadv** がディレクトリー・サーバーを見つけられるようにします。**/opt/dcelocal/var/dced/cdscache.inf** ファイルが存在しなければ、作成します。フォーマットは次のとおりです。

```
<cds_server_ip_addr> ip
```

次に例を示します。

```
9.3.53.233 ip
```

5. DCE デーモンを開始します。

```
start.dce
```

6. 以下のステップは、**/opt/dcelocal/var/dced/cdscache.inf** ファイルが存在しない場合にのみ行う必要があります。

   a. CDS アクセスがまだリストアされていないため、 **BIND_PE_SITE** 変数を設定します。

   ```
   export BIND_PE_SITE=1
   dce_login cell_admin
   ```

   b. CDS サーバーの新しい IP アドレスを **cdsadv** プロセスに通知します。

   ```
   cdscp define cached server <server_name> tower ¥
           ncadg_ip_udp:<new_address>
   ```

   c. この時点で、クライアントはサーバーの新しい IP アドレスを完全に認識します。 **dce_login** が通常どおりできることを確認します。

   ```
   unset BIND_PE_SITE
   dce_login cell_admin
   ```

7. ネームスペースをうまくアクセスできることを確認します。

```
     cdsli -o
/.:/cell-profile
/.:/fs
/.:/lan-profile
/.:/sec
```

DCE クライアントの IP アドレスの変更

1. DCE デーモンを開始します。

```
start.dce
```

2. CDS の中のクライアント・セルフ・エントリーを更新します。

```
rpccp unexport -i e1af8308-5d1f-11c9-91a4-08002b14a0fa,3.0 ¥
                 /.:/hosts/<client_name>/self
rpccp export -1 e1af8308-5d1f-11c9-91af-08002b14a0fa,3.0 ¥
                -b ncadg_ip_udp:<new_ip_addr>[135] ¥
                 /.:/hosts/<client_name>/self
```

cell オブジェクトの変更または拡張

cell タスク・オブジェクトはスクリプトとして実装されるので、管理者は、サイトごとにそれを変更または拡張することができます。行うことができる変更または拡張の例を以下に示します。

- DFS サーバー情報を表示する方法の追加。
- セル内のすべてのホストのリストを省略したり、特定の DCE サーバーだけを表示するための、 **cell show** オペレーションへのオプションの追加。

本書のガイドの 1ページの『第1部 DCE 制御プログラム』で、新しい **dcecp** オブジェクトを作成する方法、または **dcecp** 言語で書かれた既存のオブジェクトの変更方法が説明されています。

第6章 DCE ホストの管理

大きな DCE セルには多くのホスト・コンピューターを置くことができ、あるコンピューターは DCE サーバーとアプリケーション・サーバーの両方を実行し、また別のコンピューターはクライアント・システムとしてのみ実行します。さらに、アプリケーション・サーバーを実行するが、常駐ユーザーに対してはクライアントとして実行するホストもあります。 DCE ホスト構成のこのような柔軟性のために、セル内の各ホスト上で何が実行中であり、何が使用可能かを制御したり追跡するのが困難になる場合があります。 **host** タスク・オブジェクトは、ホストに関連する DCE およびアプリケーションのプロセスを示すので、管理者はマシン上の DCE サーバーおよびアプリケーションのプロセスを管理しやすくなります。

host タスク・オブジェクトを使用すれば、セル内のローカル・ホストおよびリモート・ホスト上のプロセスに関する情報を表示し、セル全体のホスト上の DCE プロセスを開始および停止することができます。また、セル内にローカル DCE ホストを構成したり、セルからリモート DCE ホストを削除 (構成解除) したりすることもできます。このオブジェクトに対するオンライン・ヘルプは、 **dcecp** の **host help** コマンドおよび **host operations** コマンドでもって使用することができます。

リモート・ホスト上で実行されるすべての **host** オブジェクト・オペレーション (**host catalog** を除く) の場合、リモート・ホスト上で **dced** が実行中でなければなりません。

セル内の DCE ホストのリスト

host catalog オペレーションを使用すれば、 DCE セル内に構成されている DCE ホストの個数と名前を判別することができます。このオペレーションは、特定のホストがセルにすでに構成されているかどうかを判別する場合に便利です。 **host catalog** オペレーションは、実際には **directory list /.:/hosts** オペレーションを実行し、ホストとは対話しないため、このオペレーションが機能するためにはホストが実行中である必要はありません。この方法は、各ホストがその名前を **/.:/hosts** ディレクトリーに登録するという規則に基づいています。ホストが名前を他のディレクトリーに登録している場合は、 **host** タスク・オブジェクトの **host catalog** オペレーションを変更する必要があります。 CDS ディレクトリーの目的および使用法の詳細は、215ページの『第18章 CDS ディレクトリーの管理』にあります。

host catalog オペレーションは、DCE サーバーを個別にリストしないことを除けば、 **cell show** オペレーションに似ています。次に示すオペレーションは、セル内に構成されているすべての DCE ホストをリストします。

```
dcecp> host catalog
/.../my_cell.goodco.com/hosts/bigbox
/.../my_cell.goodco.com/hosts/drifter
/.../my_cell.goodco.com/hosts/duh
/.../my_cell.goodco.com/hosts/heater
/.../my_cell.goodco.com/hosts/pc1
/.../my_cell.goodco.com/hosts/pc2
/.../my_cell.goodco.com/hosts/pc3
/.../my_cell.goodco.com/hosts/peewee
```

```
/.../my_cell.goodco.com/hosts/xoltar
/.../my_cell.goodco.com/hosts/xray
/.../my_cell.goodco.com/hosts/zoof
dcecp>
```

以下の例のように、**-simplename** オプションを使用すれば、セル名を省略することができます。

```
dcecp> host catalog -simplename
hosts/bigbox
hosts/drifter
hosts/duh
hosts/heater
hosts/pc1
hosts/pc2
hosts/pc3
hosts/peewee
hosts/xoltar
hosts/xray
hosts/zoof
dcecp>
```

DCE ホストに構成されているすべてのサーバーの表示

DCE サーバーおよびアプリケーション・サーバーが複数のホストに常駐している大きなセルでは、特定のホストで実行するために、どのサーバーが構成されているかを知りたい場合があります。 DCE 制御プログラムの **host show** オペレーションは、 DCE ホストのサーバー構成情報および実行情報を読み取って、そのホスト上で構成されているサーバーのリストを戻します。このリストは、各サーバーの単純名および、それが実行中であるかどうかを示します。また、このリストは、セキュリティー・サーバーがマスターであるかレプリカであるか、および DTS エンティティーがクラークであるかサーバーであるかも示します。

このオペレーションは、 **server** オブジェクト (したがって、DCE ホスト・デーモン) に基づいて、構成済みのサーバーに関する情報を表示します。サーバー制御の詳細は、117ページの『第10章 DCE アプリケーションの管理』にあります。

以下の例は、DCE ホスト **xoltar** 上で実行するよう構成されている、サーバーを示します。

```
dcecp> host show /.:/hosts/xoltar
video_clip running
dts-entity running clerk
dcecp>
```

注: DCED に登録されたサーバーだけが表示されます。 DCED に登録されている DCE サーバーは **dtsd**、 **emsd**、**pwd_strengthd**、**gdad**、および **auditd** です。

DCE ホストが実行中かどうかのテスト

DCE 通信では、クライアントがサーバーと通信する前に複数のステップを要する場合が多いので、通信障害の診断が困難になることがあります。たとえば、ホストがセルに構成済みであっても、サーバーがホスト上で実行中でなかったり、 DCE サービスが現在実行中でないことがあります。 **server ping** オペレーションを使用すれば、サーバー・プロセスが実行中であるかどうかをテストできますが、これが失

敗した場合、 DCE ホストがネットワークを介してアクセス可能であるかどうかを知る方法が必要になります。 DCE 制御プログラムの **host ping** オペレーションは、ホストの DCE サービスがネットワーク上でアクセス可能であるかどうかをテストして、アクセス可能であれば **1** を戻し、アクセス不能であれば **0** を戻します。

host ping オペレーションは、リモート・ホストの DCE デーモン (**dced**) の有無をテストします。 **dced** の目的および使用法の詳細は、103ページの『第9章 DCE ホスト・サービスとホスト・データの管理』にあります。

以下の例は、ホスト **duh** 上の **dced** がネットワーク上でアクセス可能かどうかをテストします。

```
dcecp> host ping /.:/hosts/duh
1
dcecp>
```

ホスト・オブジェクトの変更または拡張

host タスク・オブジェクトはスクリプトとして実装されているので、管理者はサイト単位でそれを変更または拡張することができます。たとえば、管理者が DFS 情報をオブジェクトに追加したい場合があります。アプリケーション・サーバーを開始または停止するための特定のコマンドへの呼び出しを追加することもできます。たとえば、**printer stop** オペレーションが便利な場合があります。

このガイドの 1ページの『第1部 DCE 制御プログラム』で、新しい **dcecp** オブジェクトを作成する方法、または **dcecp** 言語で書かれた既存のオブジェクトの変更方法が説明されています。

第7章 DCE ユーザーの管理

最も頻繁に行われる DCE 管理タスクの 1 つは、 DCE 環境内のユーザーの管理です。会社の組織変更、ビジネス・ニーズの変化、および絶えず変動する経済要因はすべて、さまざまなグループまたはオーガニゼーションの間でのユーザーの移動または出入を引き起こす原因になります。

DCE がサポートするように設計されている対象の大部分は DCE ユーザーです。DCE サービスは、一部のユーザーを認証し、権利を与えると同時に、許可されていないユーザーへはアクセスを与えません。実際、ユーザーは複雑な管理要件を持っています。ユーザー情報は、複数のサービスに広がっていて、各サービス活動の確認および制御に使用されます。ユーザー情報には、プリンシパル名、グループとオーガニゼーションの情報、アカウント情報、および CDS 内の情報が含まれます。

DCE 制御プログラムには、 DCE セル内の各ユーザー情報を管理するための個別の管理オブジェクトが用意されています。これらの個別のオブジェクトは、特定のユーザー情報に小幅な調整を行うには非常に便利ですが、セルへのユーザーの追加や削除のような反復するタスクに常時使用するには不便です。もっと簡単な方法では、 **user** タスク・オブジェクトを使用して、 DCE セル内のユーザー情報を作成、削除、および表示しやすくすることができます。このオブジェクトのオンライン・ヘルプは、 **dcecp** の **user help** および **user operations** コマンドを使用して利用することができます。

新しいユーザーの作成

DCE 環境内の各**ユーザー**とは、固有の識別 (プリンシパル名) を持つ人のことです。各プリンシパルは、少なくとも 1 つのセキュリティー・グループおよびオーガニゼーションのメンバーであり、 DCE セキュリティー・サービス・レジストリー・データベース内にアカウントを持っています。必須ではありませんが、各プリンシパルは CDS 内にディレクトリーを持つこともできます。

user タスク・オブジェクトを使用してユーザーを作成する場合、いくつかの下位オペレーションを行います。

- **user create** オペレーションは新しいプリンシパル名を作成し、そのプリンシパルをセキュリティーのグループおよびオーガニゼーションに追加します。このオペレーションを起動したときにセキュリティーのグループまたはオーガニゼーションが存在していない場合、 **-force** オプションを使用して、それらの作成を強制することができます。プリンシパル属性は、デフォルト値を想定しますが、必要に応じて、他の属性を指定することができます。すべての属性のリストは、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* にあります。

  通常、セキュリティー・グループの名前は、 DCE 環境内のさまざまなサーバーおよびデータ・オブジェクトへのユーザー・アクセスを管理する、アクセス制御リスト (ACL) の中にあります。セキュリティー・オーガニゼーションは、そのオーガニゼーションのメンバーであるすべてのプリンシパルに適用される、ポリシーを維持します。ポリシーは、アカウントの有効期間、アカウント・パスワードが期限切れかどうか、または期限切れになる時期、あるいはパスワードに非英数

字を入れることができるかどうかなどを制御します。プリンシパル、グループ、およびオーガニゼーションの管理の詳細は、 357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』にあります。

- **user create** オペレーションはプリンシパルのアカウントを作成し、ユーザーのパスワードを作成します。アカウント属性はデフォルト値を想定しますが、必要に応じて、他の属性を指定することができます。すべての属性のリストは、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* にあります。

  プリンシパルのアカウントには、グループ名と組織名、アカウントの作成と期限切れについての情報、およびチケット (プリンシパルを DCE 環境内のリソースに対して識別する) の情報など、プリンシパルに関する情報が入っています。アカウントの管理の詳細については、『アカウントの作成および保守』にあります。
- 最後に、**user create** オペレーションは、 ***/.:/users/****principalname* という名前のディレクトリーを CDS に追加します。このディレクトリーには、ユーザー固有のアプリケーション所在情報を格納することができます。このオペレーションは、 ACL エントリーをデフォルトの ACL に追加して、ユーザーにこのディレクトリーに関する **rwtci** 許可を与えます。この許可によって、ユーザーはオブジェクトとリンクを挿入できますが、ディレクトリーを削除したり、ディレクトリー上の複製を管理することはできません。さらに、ユーザーは、クリアリングハウスに対する **w** (書き込み) アクセスが与えられないかぎり、追加のディレクトリーを作成することができません。 CDS ディレクトリーの目的および使用法の詳細は、215ページの『第18章 CDS ディレクトリーの管理』にあります。 ACL および CDS ディレクトリーの詳細は、195ページの『第16章 CDS 名へのアクセスの制御』にあります。

DCE セル内に新しいユーザーを作成するには、通常、多くの許可が必要であるため、セル管理者のアカウント (または同等の特権アカウント) にログインしなければなりません。 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* には、必要となる許可の一覧が示されています。

DCE セル内に新しいユーザーを作成するには、 **user create** オペレーションを起動します。以下の例では、プリンシパル名 **P_Pestana** および同じ名前のアカウントが作成されます。 **create** オペレーションでは、他人が不在セッションを使用して無許可のアカウントを作成するのを防止するために、パスワードが必要です。ユーザーのパスワードを指定するには、 **-password** オプションも指定する必要があります。必須の **-group** オプションと **-organization** オプションは、指定されたグループとオーガニゼーションにプリンシパル **P_Pestana** を追加します。オプションの **-fullname** オプションは、他のユーザー (人) がプリンシパルを認識できるようにするために、フルネームを作成します。

```
dcecp> user create P_Pestana -fullname {Patricia Pestana} ¥
       -mypwd mxyzptlk -password change.me -group users ¥
       -organization managers
dcecp>
```

user create オペレーションへの引き数としてユーザー名のリストを指定することによって、複数のユーザーを作成することができます。しかし、この方法には、いくつかの制約事項があります。作成したすべてのユーザーは、同じ初期パスワード、グループ名、およびオーガニゼーション名を持つことになります。さらに、**uid**

属性を指定することはできません。これは各ユーザーごとに固有なものであるためです。以下の例は、パスワード **change.me**、グループ名 **users**、およびオーガニゼーション名 **staff** を使用して、複数のユーザーを作成しています。

```
dcecp> user create {R_Lee B_Joy N_Lynn D_Dee} -mypwd mxyzptlk ¥
       -password change.me -group users -organization staff
dcecp>
```

ユーザー情報の表示

ユーザーに関する属性を表示したい場合があります。たとえば、1 つまたは複数のアカウントの有効期限日を見たり、あるいはプリンシパルのフルネームを表示したい場合があります。

user show コマンドは、コマンドへの引き数として組み込まれている、ユーザーに関連する属性を戻します。属性には、プリンシパル属性と ERA、およびアカウント属性とポリシーが含まれます。情報は、次のコマンドがあたかも次に示した順序で実行されたかのように、戻されます。

- **principal show**
- **account show -all**

以下のコマンドは、ユーザー **P_Pestana** に関連するプリンシパル属性およびアカウント属性を表示します。

```
dcecp> user show P_Pestana
{fullname {Pat Pestana}}
{uid 5139}
{uuid 00001413-ad4f-21cd-8c00-0000c08adf56}
{alias no}
{quota unlimited}
{groups users}
{acctvalid yes}
{client yes}
{created /.../my_cell.goodco.com/cell_admin ¥
    1994-08-01-16:41:32.000+00:00I-----}
{description {}}
{dupkey no}
{expdate none}
{forwardabletkt yes}
{goodsince 1994-08-01-16:41:32.000+00:00I-----}
{group users}
{home /}
{lastchange /.../my_cell.goodco.com/cell_admin ¥
    1994-08-01-16:41:32.000+00:00I-----}
{organization managers}
{postdatedtkt no}
{proxiabletkt no}
{pwdvalid yes}
{renewabletkt yes}
{server yes}
{shell {}}
{stdtgtauth yes}
nopolicy
dcecp>
```

user create オペレーションへの引き数としてユーザー名のリストを指定すれば、複数のユーザーに関する情報を表示することができます。

ユーザーの削除

ユーザーがオーガニゼーションを離れた場合、おそらくそのユーザーをセルから削除しなければなりません。これを行うためには、**user delete** コマンドを使用します。このオペレーションによってレジストリーからプリンシパル名が削除され、これによって、そのアカウントが削除され、そのプリンシパルがすべてのグループおよびオーガニゼーションから削除されます。このオペレーションはまた、***/.:/users/****principalname* ディレクトリーとすべての内容を CDS から削除します。

ユーザーを削除するには、通常、セル管理者に関連する、多くの許可が必要となります。 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

以下の例のオペレーションは、ユーザー **P_Pestana** をセルから削除します。

```
dcecp> user delete P_Pestana
dcecp>
```

以下のように、 **user delete** オペレーションへの引き数としてユーザー名のリストを指定すれば、セルから複数のユーザーを削除することができます。

```
dcecp> user delete {W_Rosenberry J_Hunter P_Pestana}
dcecp>
```

外部セルの許可を持っている場合、削除するユーザーのグローバル・プリンシパル名を指定することによって、そのセルから 1 人または複数のユーザーを削除することができます。たとえば、次のとおりです。

```
dcecp> user delete /.../their_cell.goodco.com/J_Jones
dcecp>
```

ユーザー・オブジェクトの変更または拡張

user タスク・オブジェクトはスクリプトとして実装されているので、管理者はサイト単位でそれを変更または拡張することができます。たとえば、管理者が DFS 情報をオブジェクトに追加したい場合があります。その他の予想される変更を次に示します。

- ユーザー用に作成された CDS ディレクトリーの位置を変更したり、完全に削除したりする。
- さまざまなオブジェクト上に置かれているデフォルトの ACL を変更する。
- ***/.:/users/****username* ディレクトリーのマスター・レプリカが常駐するクリアリングハウスへの書き込みアクセスをユーザーに与えるために、オプションを追加する。これにより、ユーザーは自身のサブディレクトリーを作成することができます。このオプションによって、個別のプリンシパル名をクリアリングハウス ACL に追加することもできます。クリアリングハウスへの書き込みアクセスを持っているグループにプリンシパルを追加するための、より簡単な方法もあります。
- そのサイトのポリシーに一致するように、新しく作成したすべてのプリンシパルおよびアカウントに特定の属性またはポリシーを設定する。たとえば、プリンシパルを **pwd_val_type** ERA を持つように設定し、アカウントをランダム・パスワードを生成するように設定することができます。

- パスワード (後でユーザーが変更できる)、グループ、オーガニゼーション、プリンシパル・ディレクトリーなどに対するそのサイト固有のデフォルトを設定する。
- **user modify** コマンドをサポートする。このようなコマンドは、グループ情報またはオーガニゼーション情報、あるいはユーザーに関連する他のいくつかの属性を変更することができます。

このガイドの 1ページの『第1部 DCE 制御プログラム』で、新しい **dcecp** オブジェクトを作成する方法、または **dcecp** 言語で書かれた既存のオブジェクトの変更方法が説明されています。

第8章 イベント管理サービス (EMS)

EMS は、DCE セル内のイベント・サービスを管理します。 EMS では、イベントとは、イベント・サプライヤーから EMS へ、および EMS から 1 つまたは複数のイベント・コンシューマーへ転送されるデータのことです。 1 つのイベントは、1 つのイベント・ヘッダーと、イベントのタイプ固有のデータが入ったイベント属性のリストから構成されます。

EMS は、以下の 3 つの主なコンポーネントから構成されます。

- **EMS デーモン (emsd)** は、以下のことを行うサーバーです。
  - イベント・サプライヤーとイベント・コンシューマーの認証と認可を行う
  - イベント・タイプ、イベント・フィルター、およびコンシューマーのデータベースを維持する
  - 各イベント・コンシューマーをイベント・フィルター・グループに関連づける
  - 関係するコンシューマーに対するイベントの送達を確実に行う
- **イベント・サプライヤー**は、イベント・データを発行する、任意の DCE ベースのユーザー・アプリケーションです。
- **イベント・コンシューマー**は、以下のことを行うリクエスターです。
  - サポートされるイベント・タイプについて、EMS に照会する
  - 既存のフィルター名のリストを取得する
  - 各イベント・タイプに対するイベント・フィルターを作成する
  - イベント・フィルター・グループにイベント・フィルターを追加する

EMS データ構造は、以下の機能にグループ化されています。

- イベント属性
- イベント構造
- イベント・タイプ
- イベント・フィルター
- コンシューマー・データ構造
- サーバー・データ構造

イベントを EMS に経路指定する機能は、SVC および 監査サブシステムに統合されています。 EMS を使用して、SVC を通して DCE SNMP サブエージェントにイベントを通信することもできます。メッセージの経路指定についての詳細は、「DCE アプリケーション開発の手引き - コア・コンポーネント」の「SVC および監査」の項を参照してください。

DCE 管理機能には、EMS サーバー、イベント・キュー、およびイベント・ログの管理が含まれます。詳細については、以下を参照してください。

- 90ページの『EMS サーバーの起動』
- 90ページの『EMS イベントのロギング』
- 92ページの『EMS コンシューマーの管理』
- 93ページの『EMS イベント・フィルターの管理』

EMS サーバーの起動

emsd コマンドは、EMS デーモンを起動します。コンシューマーがイベントを受け取るか、またはサプライヤーがイベントを供給できるためには、その前に EMS デーモンが DCE セル内のホスト・システム上で実行中でなければなりません。

EMS デーモンは、ローカル・ホスト・マシンのプリンシパル識別 (**host/***hostname***/self**) の下で実行されます。 **emsd** を起動させるときには、DCE ホスト・デーモン (**dced**) がローカル・ホストで実行中でなければなりません。 **emsd** コマンドには、CDS アドバタイザー (**cdsadv**) も必要です。

emsd コマンドには、以下のオプション・パラメーターがあります。

-l *log_directory*
: ログ・ファイルが常駐している場所を指定します。

-q *queue_size*
: EMS によってカウントされるイベントの最大数を指定します。

-w *svc_route*
: DCE 保守容易性経路指定命令を指定します。

EMS デーモンを起動し、キュー・サイズとログの場所を指定するためには、以下のように入力します。

```
emsd -q 2048 -l /opt/dcelocal/var/ems
```

EMS デーモンを起動し、保守容易性経路指定命令を指定し、最大キュー・サイズを定義するには、以下のように入力します。

```
emsd -w NOTICE:STDOUT:- -w NOTICE_VERBOSE:STDOUT-:-
```

EMS イベントのロギング

EMS イベント・ログは、 EMS に障害が発生した場合に、イベントを格納するために使用されます。 EMS は、すべてのイベントをイベント・ログに書き込み、イベント・レコードを受け取ることになっているすべてのコンシューマーに転送された後で、そのイベント・レコードを削除します。イベント・ログは、**emsd** が実行されているマシン上のファイルに保持されます。イベントは、環境変数 **EMS_EVENTLOG_DIR** によって指定された、ディレクトリーに格納されます。ローカルのイベント・ログを調べるための API が提供されています。

emslog オブジェクトは、EMS イベント・ログを表します。 **emslog** コマンドが **dcecp** 環境で発行され、このコマンドの後に以下のサブコマンドのうちの 1 つが続きます。

help オブジェクトのヘルプ情報を戻します。

operations
オブジェクトによってサポートされるオペレーションのリストを戻します。

show イベント・ログ・ファイル内のイベントのリストを戻します。

一般的な EMS ログのヘルプ情報を表示するには、以下のように入力します。

```
dcecp> emslog help
```

EMS は、以下のように表示します。

```
help        Print a summary of command-line options.
operations  Returns the valid operations for command.
show        Returns a list of events in the event log file.
```

オブジェクトによってサポートされるオペレーションのリストを取得するには、以下のように入力します。

```
dcecp> emslog operations
```

EMS は、以下のように表示します。

```
show help operations
```

イベント・ログ・ファイル内のイベントのリストを表示するには、以下のように入力します。

```
dcecp> emslog show
EMS Displays:
--- Start of an EMS event record ---
Type: SVC:Event Id: 8d1b0b00-e9e7-11ce-8af3-10005a890435
Name Service: DCE /.../eagle_dce/hosts/hidalgod.austin.ibm.com
Description Name: EMS_Test_Producer
PID: 565 UID: 0 GID: 0
Severity: NOTICE
Arrival Time: 1995-09-08-14:06:32.970+00:00I-----
Printing 16 items
Item 1: [version] = ulong int 1
Item 2: [t] = 1995-09-08-14:06:32.970+00:00I-----
Item 3: [argtypes] = char string
Item 4: [table_index] = ulong int 0
Item 5: [attributes] = ulong int 64
Item 6: [message_index] = ulong int 389738500
Item 7: [format] = char string Test Supplier starting
Item 8: [file] = char string supplier.c
Item 9: [progname] char string EMS_Test_Producer
Item 10: [line] = ulong int 63
Item 11: [threadid] = ulong int 2
Item 12: [component_name] = char string sup
Item 13: [sc_name] = char string general
Item 14: [attribute.debug] = ushort int 0
Item 15: [attribute.severity] = ushort int 4
Item 16: [attribute.actroute] = ulong int 0
--- End of an EMS event record ---
```

EMS コンシューマーの管理

EMS コンシューマーは、イベントを受け取るために、イベント・サーバーに登録します。各コンシューマーには、名前、UUID、実行するホスト、およびフィルター・グループを形成するフィルター名のリストがあります。

コンシューマーのプロセスが異常終了した場合、システム管理者は、**emsconsumer** コマンドを使用して、そのコンシューマーを削除しなければならないことがあります。

新しいフィルターがフィルター・データベースの中に定義されている場合、管理者は、**emsconsumer** コマンドを使用して、これらのフィルターをコンシューマー・フィルター・グループに追加し、コンシューマーが受け取るイベントをさらに絞ることができます。また、コンシューマーが受け取るべきすべてのイベントを受け取っていない場合には、管理者は、**emsconsumer** コマンドを使用して、フィルターをコンシューマー・フィルター・グループから削除することができます。

DCE は、コンシューマーを管理するための、 **emsconsumer** コマンドとそれに関連するサブコマンドを提供します。このコマンドは、**dcecp** 環境で発行され、以下のサブコマンドを実行することができます。

catalog
ホスト上の EMS に登録されたコンシューマーのリストを戻します。

delete ホスト上の EMS から、登録されたコンシューマーを削除します。

help オブジェクトのヘルプ情報を表示します。

modify
指定されたコンシューマーに関連するイベント・フィルター・グループを変更します。

operations
オブジェクトによってサポートされるオペレーションのリストを戻します。

show コンシューマー・フィルター・グループの中のフィルター名のリストを戻します。

以下の許可が必須です。

- **emsconsumer catalog** および **emsconsumer show** の場合、 **/.:/***hostname***/ems-server/consumers** の **r** 許可を持っていなければなりません。
- **emsconsumer delete** の場合、 **/.:/***hostname***/ems-server/consumers** の **d** 許可を持っていなければなりません。
- **emsconsumer modify** の場合、 **/.:/***hostname***/ems-server/consumers** の **w** 許可を持っていなければなりません。

EMS に登録されたコンシューマーのリストを取得するには、以下のように入力します。

```
dcecp> emsconsumer catalog
```

EMS は、以下のように表示します。

```
{consumer1 7e383761-f41f-11ce-9051-08005acd43c6 /.:/hosts/eagle.austin.ibm.com}
{consumer1 a4c7ff26-f449-11ce-a863-10005a4f3556 /.:/hosts/eagle.austin.ibm.com}
{consumer2 283cc40c-f447-11ce-9dd3-10005a4f3556 /.:/hosts/umesh.austin.ibm.com}
```

フィルター **foo** を **consumer2** イベント・フィルター・グループに追加するには、以下のように入力します。

```
dcecp> emsconsumer modify consumer2 -add {filter foo}
```

consumer2 フィルター・グループ内のフィルター名のリストを表示するには、以下のように入力します。

```
dcecp> emsconsumer show consumer2
```

EMS は、以下のように表示します。

```
{foo2 foo3 foo4 foo5}
```

EMS イベント・フィルターの管理

EMS イベント・フィルターは、サプライヤーから受信したイベントをコンシューマーに送るべきかどうかを判断するために、 EMS がそのイベントに対して適用します。

EMS イベント・フィルターは、1 つまたは複数のフィルター式の集まりです。各フィルター式は、1 つの属性名、1 つの属性演算子、および 1 つの属性値からなります。

dcecp 環境で、関連するサブコマンドを指定した **emsfilter** コマンドを実行して、ローカル・ホスト上のイベント・フィルターを管理することができます。 **--host** オプションを指定して、別のホストにコマンドを実行することもできます。受け入れられる DCE ホスト名の形式は、 DCE 名全体 (たとえば、**/.:/hosts/jurassic.austin.ibm.com**) か、またはドメイン・ネームのついた DCE ホスト名 (たとえば、**jurassic.austin.ibm.com**) のいずれかです。

emsfilter は、以下のサブコマンドを実行します。

catalog
: EMS 内のすべてのフィルター名のリストを戻します。

delete EMS から、フィルターとそれに関連するフィルター式を削除します。

help オブジェクトのヘルプ情報を表示します。

operations
: オブジェクトによってサポートされるオペレーションのリストを戻します。

show 指定されたフィルターの中のフィルター式のリストを戻します。

以下の許可が必須です。

- **emsfilter catalog** および **emsfilter show** の場合、**/.:/***hostname***/ems-server/filters** の **r** 許可を持っていなければなりません。
- **emsfilter delete** の場合、 **/.:/***hostname***/ems-server/filters/***filtername* の **d** 許可を持っていなければなりません。

EMS デーモンによって保持されているフィルターを表示するには、以下のように入力します。

```
dcecp> emsfilter catalog
```

EMS は、以下のように表示します。

```
Filter1
Filter2
```

Filter1 という名前のフィルターと、それに関連するフィルター式を削除するには、以下のように入力します。

```
dcecp> emsfilter delete Filter1
```

オブジェクトによってサポートされるオペレーションのリストを表示するには、以下のように入力します。

```
dcecp> emsfilter operations
```

EMS は、以下のように表示します。

```
catalog delete show help operations
```

Filter2 フィルター内のフィルター式のリストを表示するには、以下のように入力します。

```
dcecp> emsfilter show Filter2
```

EMS は、以下のように表示します。

```
{event_type == SVC}
{file == file.c}
```

EMS イベント・キューの管理

EMS イベント・キューのサイズは、 **-q** オプションまたは **EMS_QUEUE_SIZE** 環境変数を使用して、 **emsd** の起動時に設定することができます。 EMS が**キュー満杯**のエラーの受信を開始したら、もっと大きいキュー・サイズを使用して、デーモンを再起動する必要があります。 **emsd** の起動についての詳細は、90ページの『EMS サーバーの起動』を参照してください。

構成 GUI を使用している場合、キュー・サイズを増やすためには、環境変数しか使用できません。

EMS デーモンの管理

EMS デーモン **emsd** が持つ責任は以下のとおりです。

- イベント・サプライヤーおよびイベント・コンシューマーに関するイベント ACL を管理する
- イベント・タイプ、イベント・フィルター、およびコンシューマーのデータベースを維持する
- 各イベント・コンシューマーにイベント・フィルター・グループを関連づける
- 関係するコンシューマーに対してイベントを確実に送達する

ems コマンドとそれに関連するサブコマンドは、 DCE ホスト上の EMS デーモンを管理します。このコマンドは、**dcecp** 環境で発行され、以下のサブコマンドを実行することができます。

catalog
現行セル内で EMS デーモンが実行中のすべてのホストのリストを戻します。

help オブジェクトのヘルプ情報を戻します。

operations
オブジェクトによってサポートされるオペレーションのリストを戻します。

show EMS デーモンの属性リストを戻します。

ems show コマンドを実行するには、 ***/.:/hostname*/ems-server** の **r** 許可が必要です。

現行セル内で実行中のすべてのホストをリストするには、以下のように入力します。

```
dcecp> ems catalog
```

EMS は、以下のように表示します。

```
/.:/hosts/eagle.austin.ibm.com
/.:/hosts/umesh.austin.ibm.com
```

オブジェクトによってサポートされるオペレーションのリストを戻すには、以下のように入力します。

```
dcecp> ems operations
```

EMS は、以下のように表示します。

```
catalog show help operations
```

EMS デーモンに対する属性のリストを表示するには、以下のように入力します。

```
dcecp> ems show
```

EMS は、以下のように表示します。

```
{eventlog_dir /opt/dcelocal/dce/var/ems}
{queue_size 5000}
```

EMS サーバーに対する許可の設定

EMS は、EMS データベース内のデータの安全な操作を提供します。これには、イベント・フィルター・データベース、イベント・タイプ・データベース、およびコンシューマー・データベース内のコンシューマーのリストが含まれます。 EMS はまた、イベント・データの安全な伝送だけでなく、サプライヤーおよびコンシューマーの認証および認可も提供します。

すべての ACL は DCE ネームスペースの中の名前と関連づけられ、 EMSD サーバーは、ジャンクションを越えてネームスペースを管理します。

```
/.:/hosts/hostname/ems-server/
```

このオブジェクトに関連づけられた ACL は、このネームスペースに登録された EMSD サーバーへのアクセスを制御します。 *ems-server* に関連する許可は、以下のとおりです。

表 1. EMSD サーバー許可ビット

許可ビット	名前	説明
c	制御	サーバー上の ACL を変更します
r	読み取り	このサーバーの属性を読み取ります
s	停止	EMS サーバーを停止します
w	書き込み	このサーバー上の属性を変更します

3 つのセキュリティー・オブジェクトが、EMS サーバー・ジャンクションの下で維持されます。ディレクトリーと、それらが表すデータベースは以下のとおりです。

event-types
イベント・タイプ・データベース

filters フィルター・データベース

consumers
コンシューマー・データベース

これらのデータベースのそれぞれに、それに関連づけられた ACL があります。

イベント・タイプ・セキュリティー管理

イベント・タイプ・データベースは、DCE ネームスペースの中で、以下の名前によって表されます。

```
/.:/hosts/hostname/ems-server/event-types
```

このオブジェクトに関連づけられた ACL は、このデータベースへのアクセスを制御します。 *event-types* に関連する許可は、以下のとおりです。

表 2. イベント・タイプ・データベース許可ビット

許可ビット	名前	説明
c	制御	イベント・タイプ上の ACL を変更します
d	削除	イベント・タイプ・スキーマを削除します
i	挿入	イベント・タイプ・スキーマを追加します
r	読み取り	イベント・タイプ・スキーマの内容を読み取ります

EMS イベント・データ・アクセスは、イベント・タイプごとに与えることができます。特定のイベント・タイプのイベント・データに対する権限は、以下にある ACL を変更することによって、与えることができます。

```
/.:/hosts/hostname/ems-server/event-types/event_type_name
```

ただし、*event_type_name* は、イベント・タイプ・スキーマの中にあるイベント・タイプ名です。 SVC イベントに対して認識される名前は、以下のようになります。

```
/.:/hosts/hostname/ems-server/events/SVC
```

event_type_name に関連する許可は、以下のとおりです。

表3. イベント・タイプ許可ビット

許可ビット	名前	説明
c	制御	イベント・タイプ上の ACL を変更します
d	削除	イベント・タイプを削除します
r	読み取り	このタイプのイベントを読み取り (消費し) ます
w	書き込み	このタイプのイベントを書き込み (供給し) ます

サプライヤー権限は EMS に送られる最初のイベントで確認され、コンシューマー権限はイベントがそのコンシューマーに送られる前に確認されます。 EMS サプライヤーおよびコンシューマーのリモート API をアクセスするために、認証 RPC が使用されます。

イベント・フィルター・セキュリティー管理

フィルター・データベースは、DCE ネームスペースの中で、以下の名前によって表されます。

```
/.:/hosts/hostname/ems-server/filters
```

このオブジェクトに関連づけられた ACL は、このデータベースへのアクセスを制御します。フィルターに関連する許可は、以下のとおりです。

表4. フィルター・データベース許可ビット

許可ビット	名前	説明
c	制御	フィルター上の ACL を変更します
d	削除	イベント・フィルターを削除します
i	挿入	イベント・フィルターを追加します
r	読み取り	イベント・フィルターのリストまたは内容を入手します

イベント・フィルターのアクセス制御は、イベント・フィルターごとに与えられます。特定のイベント・フィルターに対するフィルター・アクセスの権限は、以下にある ACL を変更することによって与えられます。

```
/.:/hosts/hostname /ems-server/filters/filter_name
```

ただし、*filter_name* は、 **ems_filter_add** の呼び出しでイベント・フィルターに与えられた、イベント・フィルター名です。

イベント・フィルターに関連する許可は、以下のとおりです。

表5. イベント・フィルター許可ビット

許可ビット	名前	説明
c	制御	イベント・フィルター上の ACL を変更します
d	削除	イベント・タイプ・フィルターを削除します
w	書き込み	イベント・フィルターの内容を変更します

コンシューマーがイベント・フィルターを作成すると、そのコンシューマーのプリンシパルは、作成されたイベント・フィルターの **dwc** 許可を自動的に受け取ります。

コンシューマー・セキュリティー管理

コンシューマー・データベースは、DCE ネームスペースの中で、以下の名前によって表されます。

```
/.:/hosts/hostname/ems-server/consumers
```

このオブジェクトに関連づけられた ACL は、このデータベースへのアクセスを制御します。コンシューマーに関連する許可は、以下のとおりです。

表 6. コンシューマー・データベース許可ビット

許可ビット	名前	説明
c	制御	コンシューマーの ACL を変更します
d	削除	コンシューマーを削除します
i	挿入	コンシューマーを追加 (登録) します
r	読み取り	コンシューマーの情報をリストします
w	書き込み	コンシューマー (そのフィルター・グループを含む) を変更します

EMS セキュリティーの初期化

EMS が構成されるときに、いくつかのセキュリティー・グループがデフォルトで作成されます。それらのグループは、**ems-admin**、**ems-consumer**、および **ems-supplier** です。デフォルトの許可は、以下のとおりです。

```
/.:/hosts/hostname/ems-server object acl
            ems-admin               crws
            hosts/hostname/self     rws
            any_other               r
  /.:/hosts/hostname/ems-server/event-types object acl
            ems-admin               cri
            ems-consumer            r
            ems-supplier            ri
            any_other               r
   initial object acl (/.:/hosts/&cont;hostname/ems-server/event-types/event_type_name)
            ems-admin               cdw
            ems-consumer            r
            ems-supplier            w
  /.:/hosts/hostname/ems-server/filters object acl
            ems-admin               crdi
            ems-consumer            ir
            any_other               r
   initial object acl (/.:/hosts/&cont;hostname/ems-server/filters/filter_name)
            ems-admin               cdw
  /.:/hosts/hostname/ems-server/consumers object acl
            ems-admin               cdrw
            ems-consumer            irwd
            any_other               r
```

これらの許可は **ems_admin** グループに対して設定されるため、作成された新しい各イベント・フィルターおよびイベント・タイプは、自動的に同じ許可を受け継ぎます。

管理者は、これらのグループのそれぞれにプリンシパルを追加して、セル内で実行中のすべての EMD に対するアクセスを与えることができます。より厳重なセキュリティーが必要な場合は、グループを対応の ACL から除去し、プリンシパルを追加してください。

第3部 DCE ホストとアプリケーションの管理

第9章 DCE ホスト・サービスとホスト・データの管理

セル全体にわたる情報を作成または維持する、DTS、CDS、および DCE セキュリティー・サービス・レジストリーのような一部のサービスは、集中化されます。これらのサービスはセル内で使用できますが、そのサーバー自体は、一般に、セル内のいくつかの特定のホストにだけ存在します。

その他の DCE サービスは広く分散され、DCE セル内の各ホストごとに常駐します。各 DCE ホストで実行される DCE ソフトウェアは、ローカルのクライアント・プログラムおよびサーバー・プログラムがリモートのクライアント・プログラムおよびサーバー・プログラムとの信頼性の高い安全な方法での対話を可能にする、必須のサービスを提供します。その結果、 DCE セル内の各ホストは、この章の最初の部分で説明するような、管理的な面を持つことになります。

各 DCE ホストは、DCE 環境内のホスト・オペレーションに必須のローカル・データを保持します。セル構成が変化した場合、あるいは DCE 機能または DCE アプリケーションを追加した場合に、このデータの一部を変更する必要があるかもしれません。この章の 2 番目の部分では、 DCE 制御プログラムを使用してこのデータへのリモート認証アクセスを獲得する方法について説明します。

DCE オペレーションが何らかの理由で失敗した場合、何が起きたかを適切な人に知らせる必要があります。 DCE の保守サービス・メッセージ機能を使用すると、メッセージの重大度レベルに応じて、エラー・メッセージを経路指定することができます。この章の最後の部分では、この機能を管理する方法について説明します。

DCE ホスト・サービス

実行時ライブラリーなどの DCE ホスト・サービスは不活性であり、ホスト上に DCE を構成した後の管理は必要ありません。しかし、他のサービスは活動プログラムです。このような活動サービスの 1 つとして、ホスト上の探索サービスとして機能するエンドポイント・マッパー があります。エンドポイント・マッパーは、ホストのエンドポイント・マップ 内のサーバー通信ポート (エンドポイント と呼ばれます) をリストします。特定のサーバーを探索するリモート・クライアントは、エンドポイント・マップに含まれている情報を戻す、エンドポイント・マッパーを照会します。エンドポイント・マッパーは、他の活動サービスとともに、*DCE ホスト・デーモン* または **dced** と呼ばれる、単一のプログラムの中に組み込まれます。一般に、DCE ソフトウェアを使用してホストが構成された後、ホスト・ブート・プロセスが **dced** プロセスを、他のデーモンまたはプロセスとともに、開始します。しかし、このデーモンを手動で起動または再起動しなければならない場合があります。

dced プログラムは 1 組の DCE ホスト・サービスで構成されており、それらはホスト・システム上の DCE クライアント・アプリケーションおよびサーバー・アプリケーションの多くのニーズを満たします。

- エンドポイント・マッパー・サービスは、ホスト上で実行するサーバーのディレクトリーとして機能します。クライアントは、登録済みサーバーの通信エンドポイントを、ホストのエンドポイント・マップを調べることによって、獲得できます。

- セキュリティー検証サービスは、ローカル・ホスト上の DCE セキュリティーを管理します。
- サーバーの構成および実行サービスは、管理者がリモートでサーバーの開始条件と停止条件を設定し、明示的に各サーバーを開始および停止し、実行中のサーバーの状態をモニターできるようにします。
- キー管理サービスは、管理者がサーバーのパスワードをリモートで管理できるようにします。
- hostdata サービスは、管理者がホスト上のファイルに格納されているデータをリモートで管理できようにします。管理者にとって、このサービスは、ホストのセル名およびセル別名情報をリモートで管理する場合に非常に便利な手段です。
- 属性スキーマ機能は、管理者が新しい属性をサーバー構成情報に追加できるようにします。

通常、DCE サーバー (たとえば、DCE セル・ディレクトリー・サーバー) のホストであるシステム、あるいは認証を使用する DCE ベースのアプリケーション・サーバーまたはアプリケーション・クライアントを実行するシステムは、**dced** プロセスも実行しなければなりません。

dced プログラムが何らかの理由で失敗した場合、そのすべてのコンポーネント・サービスもダウンし、このホストはクライアントの要求に応答できなくなることは明らかです。同様に、コンポーネント・サービス (たとえば、キー管理サービス) のうちのいずれかの失敗が、何らかの理由で **dced** プログラムが予期せずに終了したことが原因の場合があります。問題が発生した場合には、 **dced** とそのコンポーネント・サービスの間のこの関係を思い出す価値があります。

DCE ホスト・サービスの開始および停止

dced プロセスは、通常、ホスト・ブート・プロセスの一部として開始しますが、場合によっては、プロセスを手動で開始する必要がある場合があります。

dced を開始するためには、クライアント / サーバー通信が依存する、根幹のすべてのネットワーク・サービスが使用可能でなければなりません。たとえば、大部分の UNIX システムでは、ネットワーク・インターフェースと経路指定サービスが使用可能でなければなりません。このようなトランスポート層サービスが確立されると、**dced** を開始することができます。 **dced** が開始された後、RPC ベースのサーバーを開始することができます。

エンドポイント・マッパーは、クライアントのサービス要求を、特権通信ポートまたは予約通信ポート (ウェルノウン・エンドポイント) で listen します。その結果、**dced** は特権ユーザーとして開始されなければなりません。

dced の一部 (エンドポイント・マップ) には、ホスト・システム上のサーバーを見つけるためにクライアントが使用する、情報が入っています。 **dced** プロセスは、この情報のコピーを *dcelocal***/var/dced/Ep.db** という名前のデータベース・ファイル内に維持するため、何らかの理由で **dced** を停止してから再び開始した場合も、情報が失われることはありません。 *dcelocal***/var/dced/Srvrexec.db** という、別のデータベース・ファイルは、ホスト上で現在実行中のサーバーに関する情報 (たとえば、各サーバーのプロセス ID) を維持します。システムをリブートした場合、この

両方のデータベース内の情報は古くなります。その理由は、大部分のサーバーは開始するたびに、異なるエンドポイントと異なるプロセス ID を与えられるためです。

dced を、ホストがブートするたびに開始するように構成することができます。これを行うには、**config.dce** で **-autostart yes** オプションを使用します。他のいずれかの DCE デーモンが開始されると、 **Start.dce** が **dced** を開始します (軽量クライアントが構成されており、かつ **dced** がない場合を除く)。

一般に、**dced** をシェルで開始する必要はありませんが、そうする必要がある場合は、**root** としてログインし、以下のコマンドを入力してください。

```
# dcelocal/bin/dced
```

デフォルトでは、**dced** は、このプログラムを実行しているホストがサポートしているトランスポートごとに 1 つのエンドポイントで listen します。つまり、ホストが TCP/IP と UDP/IP の両方のトランスポートをサポートする場合、**dced** は、クライアント要求に対して 1 つの TCP ソケットと 1 つの UDP ソケット上で listen します。オプションの *protseq* 引き数を使用すると、**dced** が使用するトランスポートを、指定したものに制限することができます。デバッグ機能として使用されるこの機能は、慎重に使用する必要があります。トランスポートを制限すると、クライアントは、除外されたトランスポート上にサーバーを置くことができなくなり、サーバーは、除外されたトランスポートを使用してサーバー自体をエンドポイント・マップに登録することができなくなります。オプションの *protseq* 引き数については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

DCE クライアント・アプリケーションまたはサーバー・アプリケーションが動作しているすべてのホスト上でホスト・サービスを実行する必要がありますが、次のように、これを実行しなくてもよい場合があります。

- 認証を行わない DCE クライアント
- 認証を行わず、しかもエンドポイント・マッパーもその他の活動状態の DCE ホスト・サービスも使用しない、DCE サーバー

DCE ホスト・サービスを開始した後、DCE 制御プログラム (**dcecp**) を使用して、すべての DCE ホストとサーバーの管理タスクを実行することができます。制御プログラムは、ホスト管理機能およびサーバー管理機能への安全なリモート・アクセスを提供します。これはすべての DCE ホストを、各ホストにログインせずに、管理できることを意味します。 1ページの『第1部 DCE 制御プログラム』では、**dcecp** を対話モードで使用する方法、および DCE 活動を管理する **dcecp** スクリプトを作成する方法について説明します。この章あるいはこの資料の他の部分で説明する管理タスクを実行する前に、これらの基本的な事項を理解しておく必要があります。

DCED ホスト・デーモンの異常終了

DCE デーモンが予期せずに停止または終了した場合、再起動させることができます。再起動された **dced** プロセスが、以前に登録されたサーバー・バインディングを失うことはありません。単に、**Ep.db** および **Srvrexec.db** ファイルから情報をロードするだけです。一般的には、**dced** を停止して再起動することはお勧めしません。理由は、セキュリティー検証サービスも停止されるからです。

dced が停止するか予期せずに終了した場合は、一部のファイルが壊れている可能性があるため、 DCE を再起動する前にそれらを除去してから再作成する必要があるかもしれません。そのような場合、以下のようにしてください。

1. コマンド行に **stop.dce all** と入力し、 DCE コンポーネントに対して構成したすべてのデーモンを停止します。
2. **clean_up.dce** と入力し、壊れている可能性のあるファイルを除去します。除去したファイルはどれも、必要に応じて再作成されます。
3. **start.dce all** と入力し、すべての構成済みコンポーネントを再起動します。

ホスト・データの管理

DCE セル内の各ホストは、 DCE 環境での操作に必須のローカル・データを維持管理します。たとえば、各ホストの DCE 識別は、ホストのホスト名、セル名、およびセル別名を指定する、特定のデータ項目に基づきます。現在、これらのデータ項目は、 *dcelocal***/dce_cf.db** と呼ばれる、ローカル・ファイルに格納されています。これらを含めて、データ項目は、 DCE 制御プログラムの **hostdata** オブジェクトを使用して、リモートで変更することができます。

hostdata オブジェクトにもさまざまな用途があります。このオブジェクトによって、安全で、プラットフォームから独立した方式でもってリモート・ホスト上の汎用のデータとファイルをアクセスすることが可能となるため、管理者にとってはきわめて便利です。この章の最後の部分では、この強力なアクセス方式について調べます。

ホスト・データへのアクセスの許可

アクセス制御リスト (ACL) は、無許可のプリンシパルが hostdata 情報を作成、変更、または削除することを防ぎます。 2 種類の ACL によって hostdata 情報が保護されます。 ACL の 1 つのタイプは、hostdata 項目が常駐しているコンテナーを保護します。 2 番目のタイプは、個々の hostdata 項目を保護します。

ここでは、hostdata 情報を保護する、ACL の管理方法について説明します。 ACL 保護の設定および使用の詳細については、 329ページの『第28章 アクセス制御リストの使用』を参照してください。

hostdata コンテナーの許可

DCE では、hostdata 項目は、実際には **dced** によって維持される補助記憶メカニズムである、コンテナー に入っています。 UNIX システムでは、通常、これは *dcelocal***/var/dced/Hostdata.db** というファイルです。このファイルは root によって所有され、そして **dced** を介するそのアクセスは ACL によって保護されます。これらの ACL 許可は、コンテナー内のデータにアクセスできる人を制御します。各 DCE ホストには、以下の名前の hostdata コンテナー ACL が 1 つあります。

/.../*cellname***/hosts/***hostname***/config/hostdata**

hostdata コンテナー ACL は、以下の許可を持っています。

c (制御)
コンテナー ACL を変更します。

r (読み取り)
コンテナー内の hostdata 項目のリストを読み取ります。

i (挿入)
新しい hostdata 項目を作成します。

I (挿入)
I 許可が存在しますが、それは hostdata 項目には適用されません。この許可はサーバー制御機能に適用されますが、これについては、117ページの『第10章 DCE アプリケーションの管理』で説明します。

ACL を表示または変更するには、**dcecp acl** オブジェクトを使用します。たとえば、以下のオペレーションを使用すれば、ホスト **silver** 上の hostdata コンテナー・オブジェクトの ACL を表示することができます。

```
dcecp> acl show /.:/hosts/silver/config/hostdata
{user hosts/silver/self criI}
{unauthenticated r}
{any_other r}
dcecp>
```

hostdata 項目の許可

次の各ホスト識別データ項目が、ACL によって保護されます。

/.../*cellname***/hosts/***hostname***/config/hostdata/host_name**
/.../*cellname***/hosts/***hostname***/config/hostdata/cell_name**
/.../*cellname***/hosts/***hostname***/config/hostdata/cell_aliases**
/.../*cellname***/hosts/***hostname***/config/hostdata/post_processors**

各 ACL は、以下の許可を持つことができます。

c (制御)
ACL を変更します。

d (削除)
項目を削除します。

p (除去)
項目に関する補助記憶域を削除します。

r (読み取り)
項目のデータを読み取ります。

w (書き込み)
項目のデータを変更します。

ACL を表示または変更するには **acl** オブジェクトを使用します。たとえば、以下のオペレーションを使用すれば、ホスト **silver** 上の **cell_aliases** hostdata 項目の ACL を表示することができます。

```
dcecp> acl show /.:/hosts/silver/config/hostdata/cell_aliases
{unauthenticated ---r-}
{user hosts/silver/self cdprw}
{any_other ---r-}
dcecp>
```

ホスト・セル名情報の変更

hostdata オブジェクトを使用すれば、DCE ホスト上でデータ項目を追加、変更、および削除することができます。管理者にとっては、このオブジェクトはホストのセル名またはセル別名情報の変更に便利であり、ホストでアクセス可能な他のデータについて操作することもできます。

各 DCE ホストは、ホストが登録されているセルのセル名またはセルの別名の保護ローカル・コピーを保持します。ホストは、**root** が所有する */dcelocal/***dce_cf.db** というローカル・ファイルにこの情報を保持します。ホストは、認証の目的のために (ホスト識別情報の一部として) この情報を使用します。

ホスト・セル名情報はかなり安定しているのが普通ですが、次の場合のように、この情報を変更しなければならない場合があります。

- ホストが別のセルに移動する場合
- ホストのセル名が変更されるか、あるいはセル名が別名を獲得する場合

しかし、上記の状態のいずれかが発生した場合、通常は、ホスト上のセル名情報を更新するだけでは不十分です。セル名情報は、 CDS 内および DCE セキュリティー・サービス・レジストリー内でも更新しなければなりません。この目的のために、**dcecp** では、変更しなければならないセル名情報の更新を行う、 **cellalias** タスク・オブジェクトを提供しています。

ホストが別のセルに移動する場合、 **unconfig.dce** を実行してそのホストを 1 つのセルから除去する必要があります。その後、**config.dce** を実行して新しいセルにそのホストを設定します。

ホストのセル名が変更されるか、またはセル名が別名を獲得する場合には、**cellalias** オペレーションを実行する必要があります。このオペレーションは、CDS 内、DCE セキュリティー・サービス・レジストリー内、およびセルの中の影響を受ける各ホストの **dce_cf.db** ファイル内のセル情報を更新します。

しかし、高位の **dcecp** タスク・オブジェクトでは、破壊されたファイルの修正や何らかの理由でホストを手動で構成する場合に必要となるような、十分な制御を提供しない場合があります。このような場合には、**hostdata** オブジェクトを使用して、個々のホスト上のセル名情報を変更することができます。

ただし、**hostdata** オブジェクトをこのように使用する主な目的は、ホストのセル情報が、DCE レジストリーまたは CDS に格納されている、他のセル情報と同期していない場合に使用する、トラブルシューティング・オペレーションです。この状態は、多数のホストがあるセル内ではよく起きることがあります。

ホスト上のセル名またはセル別名情報を更新する場合には、**hostdata** を使用してください。以下の例は、 **/.../my_cell.goodco.com** というセル内の **hostdata** オブジェクトをカタログしています。その後、ホスト **silver** 上の *cell_name* オブジェクトの内容を表示します。最後に、ホスト **silver** 上で、セル名を **/.../my_cell.goodco.com** に変更します。

```
dcecp> hostdata cat
/.../my_cell.goodco.com/bronze/config/hostdata/dce_cf.db
/.../my_cell.goodco.com/bronze/config/hostdata/cell_name
/.../my_cell.goodco.com/bronze/config/hostdata/host_name
```

```
/.../my_cell.goodco.com/bronze/config/hostdata/cell_aliases
/.../my_cell.goodco.com/bronze/config/hostdata/post_processors
dcecp> hostdata show cell_name
{uuid 00174f6c-6eca-1d6a-bf90-0000c09ce054}
{annotation  {Name of cell}}
{storage cell_name}
{data {/.../old_cell.goodco.com}}
dcecp> hostdata modify ¥
       /.../my_cell.goodco.com/hosts/bronze/config/hostdata/cell_name ¥
       -data {/.../my_cell.goodco.com}}
dcecp>
```

他のホスト・ファイル内のデータの操作

hostdata オブジェクトはセル名およびセル別名の情報を変更する場合に便利ですが、他にもさまざまな用途があります。たとえば、DCE ホスト上でアクセス可能なすべてのファイルのデータを追加、変更、および削除することができます。

便利な例の 1 つは、新しい CDS 属性の追加です。各 DCE ホストには、それ独自の CDS 属性ファイル (**cds_attributes**) があり、そこには各 CDS 属性のオブジェクト ID が保管されています。ローカル・ホストのエディターを使用すれば、属性を追加してから、新しいファイルを各ホストにコピーすることができます。しかし、この方法の場合、各ホストへのログインが必要になります。もっと簡単な方法では、 **hostdata** オブジェクトを使用して、新しい属性を CDS 属性ファイルに追加する方法です。オペレーションを **foreach** ループ内に置いて、セル内の各ホストについてそれを再実行します。

1. **hostdata** オブジェクトの 1 つのオブジェクトとして、 CDS 属性ファイルをアクセス可能にします。まず最初に、**hostdata** オブジェクトを使用して、 CDS 属性ファイルを表す CDS エントリーを作成します。 storage 属性を CDS 属性ファイルのホスト・ファイル名に設定します。以下の例では、CDS 属性ファイルがデフォルトの位置にあり、そのファイルが存在していることを前提としています。

   ```
   dcecp> hostdata create /.:/hosts/silver/config/hostdata/cds_attr ¥
   > -storage /opt/dcelocal/etc/cds_attributes -entry
   dcecp>
   ```

2. **hostdata** オブジェクトは、ファイル内のすべてのデータを、指定された新しいデータと置き換えることによって、ファイル内のデータを変更します。次の例は、これを行うための 1 つの方法を示しています。まず最初に、すべての行を変数内の **dcecp** リスト要素として検索して格納します。次に、**attrlist** コマンドを使用して新しい変数を作成し、新しい行をリスト要素として、その変数に追加します。最後に、新しい変数を元のファイルにコピーします。

   ```
   dcecp> set val [attrlist getvalues [hostdata show /.:/hosts/silver/config/ ¥
   hostdata/cds_attr] > -type hostdata/data]
   dcecp> set newval [attrlist add $val -member {NEW_ATTR 1.2.3.4}]
   dcecp> hostdata modify /.:/hosts/silver/config/hostdata/cds_attr -data
   $newval
   dcecp>
   ```

保守サービス・メッセージの経路指定

DCE 保守サービス・メカニズムは、主にサーバー通知メッセージおよびエラー・メッセージ、すなわち、サーバーの保守および管理 (最も広い意味で) に関係する人にとって重要なメッセージのために使用されることを意図しています。このメカニズムの基本的なアイデアは、通常のオペレーションの維持または復元のために重要な、すべてのサーバー・イベントを、内容を読めば分かるメッセージで報告することです。その結果、(すべてのイベントが正しく識別され報告されていることを前提とすると) ユーザーおよび管理者は特定の状態でどのアクションをとるべきかを常に認識することができるようになります。

注: ユーザーがプロンプトとして出した対話式の、クライアントが生成したメッセージは、標準 DCE メッセージ・インターフェースを介して処理されます。

保守サービス・コンポーネントは、DCE コンポーネント (RPC、DTS、セキュリティーなど) によって、それら自身のサーバー・メッセージのために使用されます。また、この保守サービス・コンポーネントは、アプリケーションのサーバー・メッセージを標準化したい DCE アプリケーション・プログラマー用の API として提供されます。 (保守サービス API については、*IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* に説明があります。)

メッセージングでは、XPG4 メッセージ・カタログ (*X/Open Portability Guide* 参照) を使用してメッセージ・テキストを保持しますが、それにより XPG4 機能に層が 1 つ追加されます。メッセージ・カタログおよびその他の必要なデータ (および資料) のファイルは、 **sams** (symbols and message strings (シンボルおよびメッセージ・ストリング)) というユーティリティーによって生成されます。メッセージを使用するプログラムについての編成情報を確立するテキスト・ファイルの後に、メッセージ自体の一連の仕様を続けたものを入力します。保守サービス・メカニズムを使用すると、システム管理者はこうしたメッセージの経路指定を制御することができます。具体的には、メッセージに定義されている重大度レベル (**FATAL**、**ERROR** など) にもとづいて、メッセージ経路指定を定義することができます。

以下の項では、保守サービス・メッセージの経路指定を制御する方法について説明します。最初に、DCE の保守サービス・メッセージの概要を説明します。次に、メッセージ重大度レベルを使用して経路指定を制御する方法について解説します。最後に、保守サービス・メッセージの経路を指定する、いろいろな方法について説明します。

保守サービス・メッセージの重大度レベル

保守サービス・メッセージは、*重大度レベル* によって類別されており、この重大度レベルはプログラムにメッセージを出させる状態についての重要な情報を提供します。どのメッセージの重大度も、メッセージ自体のテキスト内で定義されるので (たとえば、**NOTICE** は、メッセージが通知であることを示します)、システム管理者は重大度レベルに応じてメッセージを経路指定することができます。

111ページの表7 は、可能な重大度レベルを挙げ、それぞれのレベルについて説明したものです。

表 7. 保守サービス・メッセージ重大度レベル

名前	意味
FATAL	致命的エラーで終了しました。回復不能エラー (たとえば、データベースの破壊) が起きたため、おそらく訂正には手操作による介入が必要になります。通常、プログラムはこのエラーの直後に終了します。
ERROR	エラーが検出されました。非端末の (たとえば、タイムアウト)、あるいは人の介入により訂正可能な、予期しないイベントが発生しました。プログラムはオペレーションを続けますが、一部の機能またはサービスが使用できない場合があります。この重大度レベルは、特定の要求またはアクションを完了することができないことを示す場合にも使用されます。
WARNING	訂正可能なエラー。エラーが起きましたが、自動的に訂正されました (たとえば、構成ファイルが検出されなかったので、デフォルト値が代わりに使用された場合)。この重大度レベルは、結果が好ましくない (たとえば、空でないディレクトリーを削除した結果、すべてのファイルが削除される場合) エラーになる*可能性*のある 状態を示す場合にも使用されます。この重大度レベルは、訂正されないとエラーになる状態 (たとえば、プリンターの用紙が無くなりそうな場合) を示す場合にも使用されます。
NOTICE	通知。たとえば、サーバーが始動したなど、重大なルーチンの主要イベントが発生しました。
NOTICE_VERBOSE	冗長な通知。たとえば、ディレクトリー・エントリーが削除されたなど、重大なルーチン・イベントが発生しました。

保守サービス・メッセージの経路指定方法

保守サービス・メッセージは、通常の出力宛先へ書き込むことができます。次の 4 つの方法のいずれかで、保守サービス・メッセージを経路指定することができます。

- **dcecp log** オブジェクトによる方法 (サーバーがリモート保守サービス・インターフェースをサポートする場合)
- 経路指定ファイルの内容による方法
- 環境変数の内容による方法
- コマンド行フラグ (通常、**-w**) による方法 (サーバーによってサポートされる場合)

注: これらの方法はいずれも、保守サービス経路指定用のストリング構文形式を受け入れます。さらに、**dcecp** では、 Tcl (ツール・コマンド言語) 構文も使用することができ、これを使用すると、スクリプトの作成が容易になります。

メッセージの*経路指定* は、実際には、以下の 2 つの指定から構成されます。

- メッセージの処理方法 (つまり、メッセージが書き出される形式)
- メッセージの送信先 (宛先)

この 2 つの指定は密接に関連している場合もあれば、特定の宛先を指定すると、メッセージを特定の形式で書き出さなければならないことを暗黙に指定することになる場合もあります。これにより、特定の組み合わせを省略することができます。

保守サービス・メッセージの経路指定方法については、以下の項で個別に説明します。

dcecp log オブジェクトの使用

dcecp log オブジェクトは、特定のサーバーについて DCE 保守サービス・メッセージの経路指定の現在の状態を表します。 **log** オブジェクトは、保守サービス経路指定とデバッグ経路指定の両方をサポートします。

log オブジェクトは、多数のオペレーションをエクスポートします。保守サービス・メッセージの経路指定には以下のオペレーションが便利です。

- **log list** オペレーションは、サーバーによって登録されたコンポーネントのリストを戻します。 **-comp** オプションを使用すると、指定された 1 つまたは複数のコンポーネントについて副コンポーネントのリストも戻すことができます。
- **log show** オペレーションは、サーバーに対する現行の保守サービス経路指定を記述するリストを戻します。
- **log modify** オペレーションは、指定された 1 つまたは複数のサーバーに対して、メッセージ経路指定を設定します。

dcecp log オブジェクトおよびそのサポートされるオペレーションの構文の詳細については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

この節の残りの部分では、**log modify** オペレーションおよび、それを使用して保守サービス・メッセージの経路を確立する方法だけについて説明します。経路指定は、常にサーバーごとに設定され、各サーバーの **log** オブジェクトの中に記録されることに注意してください。

log modify オペレーションの構文は、以下のとおりです。

```
log
modify {string_binding_to_server | RPC_server_namespace_entry} ¥
    {-change serviceability_routing_specifications}
```

複数のターゲット・サーバーを、スペースで区切ったリストとして指定することができます。サーバーのネットワーク位置を記述する RPC ストリング・バインディング (*string_binding_to_server*) か、サーバーのネームスペース・エントリー (*RPC_server_namespace_entry*) のいずれかを指定することによって、各サーバーを指定します。複数のサーバーを指定する場合、同じリスト内で複数の形式を混合することができます。

serviceability_routing_specification は、スペースで区切られた、保守サービス経路指定要素のリストです。個々の経路指定要素の指定の中では、スペースを使用することはできません。各経路指定要素は、以下に示すように (ストリング構文形式で表示されています)、ポータブル文字セット (PCS) データを含んだ 4 つのフィールドで構成される、部分ストリングです。

severity:*output_form*:*destination*[:*application-defined*]

ここで、

severity

メッセージ重大度レベル: **FATAL**、**ERROR**、**WARNING**、**NOTICE**、または **NOTICE_VERBOSE**。

output_form

関連した重大度レベルのメッセージの処理方法を示します。次のいずれかでなければなりません。

BINFILE

これらのメッセージを 2 進数のログ・エントリーとして書き込みます

TEXTFILE

これらのメッセージを人間が読むことができるテキストとして書き込みます

FILE **TEXTFILE** と同じです

DISCARD

これらのメッセージを記録しません

STDOUT

これらのメッセージを人間が読むことができるテキストとして、標準出力に書き込みます

STDERR

これらのメッセージを人間が読むことができるテキストとして、標準エラーに書き込みます

NOTICE メッセージまたは **NOTICE_VERBOSE** メッセージ (あるいはその両方) を **STDERR** に経路指定しないでください。これらのメッセージは、**STDOUT**、**FILE**、**EMS** などに経路指定する必要があります。**NOTICE** メッセージまたは **NOTICE_VERBOSE** メッセージ (あるいはその両方) が **STDERR** に経路指定されると、DCE の構成は失敗します。

BINFILE として書き込まれたファイルは、 1 組のログ・ファイル API (*IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Directory Services* に説明があります) を使用して、読み取りおよび操作を行うことができます。

DCE の構成時は、情報的なメッセージを **STDERR** に経路指定しないでください。構成コードがそれらのメッセージを誤ってエラーとして解釈してしまう可能性があるからです。構成時は、それらのメッセージは他のいずれかのファイルに経路指定する必要があります (デフォルトの経路指定ファイルではそのようになっています)。

BINFILE、**TEXTFILE**、および **FILE** *output_form* 指定子の後に、以下の形式の 2 桁の指定子がつく場合があります。

```
.gens.count
```

ここで、

gens 保持しなければならないファイルの数 (すなわち世代) を指定する整数

count 各ファイルに書き込まなければならないエントリー (すなわち、メッセージ) の数を指定する整数

指定された単純名に **.** (ドット) を追加し、その後に現行世代番号を続けることによって、複数のファイルが命名されます。 1 つのファイル内のエントリーの数が *count* で指定された最大値に達すると、そのファイルはクローズされ、世代番号が増やされて、次のファイルがオープンされます。

最大数のファイルが作成され、いっぱいになると、世代番号は **1** にリセットされ、その番号を持つ新しいファイルが作成され書き込まれ (したがって、同じ名前を持つ既存のファイルが上書きされ)、さらに次の番号のファイルが上書きされ ... というようになります。このようにして、プログラムがメッセージを書き込み続ける限り、メッセージは書き込み続けられますが、ファイルは最初のファイルに循環するため、ログ・ファイルの合計数が *gens* を超えることはありません。

destination

メッセージの送信先を指定します。これはパス名です。指定された *output_form* が **DISCARD**、**STDOUT**、または **STDERR** である場合、このフィールドをブランクのままにすることができます。このフィールドには、ファイル名の中に **%ld** ストリングも入れることができます。これは、そのファイルが作成される時点で、メッセージを書き込んだプログラムのプロセス ID により置き換えられます。ファイル名には、: (コロン)、; (セミコロン)、% (パーセント記号)、またはスペース文字を含めることができません。

application-defined

アプリケーション固有の情報に使用されます。標準 DCE プログラムは、これを無視します。

ストリングの構文: 保守サービス経路指定のストリング構文は次のとおりです。

severity:*output_form*:*destination*[:*application-defined*][;...]

複数の経路指定を、セミコロンで区切られたリストとして定義することができることに注意してください。

たとえば、

```
FATAL:TEXTFILE:/dev/console;STDOUT:
ERROR:TEXTFILE.5.100:/tmp/errors
EXIT:DISCARD:
*:FILE:/tmp/svc-log
NOTICE:BINFILE:/tmp/log%ld
WARNING:STDOUT:
```

上記の指定は、保守サービス・メカニズムに対して、以下のことを行うよう指示します。

- 致命エラー・メッセージをコンソールおよび標準出力に送信する
- その他のエラー・メッセージを、ログを記録するファイルに送信する
- 通常の終了レポートを廃棄する

- すべてのメッセージをログ・ファイルに書き込む
- 通知メッセージを一時的 2 進ログに送信する
- 警告メッセージを標準出力に送信する

Tcl の構文: 保守サービス経路指定のための Tcl の構文は、以下のとおりです。

{*severity output_form destination application-defined*}

なお、*severity*、*output_form*、*destination*、および *application-defined* は、前に説明したとおりに指定します。 Tcl の構文では、複数の経路指定は、以下の形式になります。

{ {*specification*} {*specification*} {*specification*} }

たとえば、前に説明したストリング形式の場合の指定例は、 Tcl の構文では、以下のように表されます。

```
{FATAL { {TEXTFILE
/dev/console} STDOUT} }
{ERROR TEXTFILE.5.100 /tmp/errors}
{EXIT DISCARD}
{* FILE /tmp/svc-log}
{NOTICE BINFILE /tmp/log%ld }
{WARNING STDOUT {} }
```

経路指定ファイルの使用

*dce-local-path***/var/svc/routing** というファイルが存在する場合、このファイルの内容 (適切な形式の場合) は、保守サービス・メカニズムによって書き込まれるメッセージの経路指定を判別するために使用されます。

保守サービス経路指定ファイルのデフォルトの場所は、通常、*/opt/dcelocal***/var/svc/routing** です。ただし、環境変数 **DCE_SVC_ROUTING_FILE** の値を希望する完全パス名に設定することにより、これとは異なるファイルの場所を指定することができます。

経路指定ファイルには、 (メッセージ重大度レベルに応じた) いろいろな種類のメッセージに必要な経路を指定する行が入っています。各行は、次のような 3 つのフィールドで構成されています。

severity:*output_form*:*destination*[:*application-defined*][...]

追加の *output_form*:*destination* のペアをセミコロンで区切ったリストとして指定することによって、複数の経路指定を行うことができます。

経路指定ファイルの中では、# 文字で始まるブランク行は、コメントとして扱われます。

環境変数の使用

保守サービス・メッセージ経路指定は、特定の環境変数の内容によっても指定することができます。環境変数を使用した場合、経路指定ファイルによって指定された、競合するすべての経路指定は、その指定した経路指定によって指定変更されます。

経路指定は、希望する経路指定を以下の環境変数に指示することによって、 (重大度レベルに応じて) 指定することができます。

- **SVC_FATAL**
- **SVC_ERROR**
- **SVC_WARNING**
- **SVC_NOTICE**
- **SVC_NOTICE_VERBOSE**
- **SVC_BRIEF**

各変数には、1 つのストリングが以下の形式で含まれていなければなりません。

severity:*output_form*:*destination*[:*application-defined*][...]

追加の *output_form*:*destination* のペアをセミコロンで区切ったリストとして指定することによって、複数の経路指定を行うことができます。

第10章 DCE アプリケーションの管理

DCE の発展に伴って、共通して必要とされる機能が DCE の下部構造に組み込まれています。例を挙げると、 DCE にはサーバー制御機能が組み込まれているので、サーバー・オペレーションを管理し、サーバーの終了を効率的に行うことができます。アプリケーションの開発者は、このような機能を使用すれば、各機能を個別に処理する特別なメカニズムを各サーバー内に実装する必要がありません。

共通して必要とされる機能をアプリケーションから DCE の下部構造に移動することには、重要な利点があります。アプリケーションが小さくなり、開発と保守が容易になります。さらに重要な利点として、アプリケーションが多くの特別なコードを必要としなくなるので、再構成、および各種のクライアントとの再接続が容易になります。この適応性は、変化するビジネス要件にオーガニゼーションが対応するために重要なことです。

DCE アプリケーションは常に管理的な側面を持っています。プログラムには、それ自身の固有の要件を管理するために必要な機能が組み込まれていることが多いですが、この方法は管理者にとって、不便であり、柔軟性に欠けます。今や、事実上、すべての管理機能が、**dcecp** を通して、プログラマーおよび管理者に等しく使用可能となりました。しかし、これは、プログラマーが、これらの問題を処理する必要がなくなったということではありません。クライアント・プログラムとサーバー・プログラムを指定の条件のもとで開始および停止するように構成された、**dcecp** でもって作成されたスクリプトを、一部のプログラマーが提供することが期待されています。

この方法は、アプリケーションを管理するための便利で一貫した方法を提供しますが、プログラミング上および管理上の問題がオーバーラップする領域も生じます。この章での説明では、管理アクションが必要な状況を示しながら、このオーバーラップする領域について説明します。

サーバー・オペレーションの制御

DCE アプリケーション・サーバーに関する従来の概念は、サーバーが実行中であり、クライアントによるサービスの要求を待機中であることを想定しています。これは、いくつかの一般的なサーバー・オペレーションには効果的なモデルですが、DCE アプリケーションで必要とする柔軟性がありません。商用環境では、さまざまな種類のサーバーが存在することが予想されます。常時使用可能でなければならないサーバーや、 1 日の特定の時間に限って必要とされるサーバーもあります。あるいは、めったに使用されないサーバーや、使用時期の予想が不可能なサーバーもあります。

アプリケーション・プログラマーや管理者は、必要な場合にサーバーを自動的に起動または再起動し、サーバー・オペレーションをモニターするようなスクリプトあるいはアプリケーションを作成することにより、この種の問題を解決することができます。このような解決方法は、多くの場合、起動プログラムおよび遮断プログラムのようなホスト・ユーティリティー、あるいは **cron** のようなスケジューラーに依存しています。しかし、そのためにはしばしば、管理者が、各ホスト上の個別のシステム管理アカウントにログインすることが必要になります。さらに、この方法

では、特に異種環境では、ポータブルでない可能性がある装置独立のサーバー制御メカニズムに関して、開発者と管理者の負担がさらに大きくなります。

DCE は、DCE アプリケーション・サーバーを制御するための、さまざまな方法を提供する、サーバー制御機能を用意することによって、これらの問題のいくつかを解決します。サーバー制御機能は DCE デーモン (**dced**) の一部であるため、サーバーは、**dced** が実行するところであれば、どこでもこれを使用することができます。さらに、この制御機能の管理機能は **dcecp** を介してアクセス可能であるため、管理者は一貫した (ポータブルな) 方法を使用して、 **dcecp** が使用可能などのホストからでもサーバーを管理することができます。さらに、サーバー制御機能に対するアクセスの認証が行われるので、サーバー制御情報が無許可で、あるいは偶然に、不正に変更されることがなくなります。

以下の節では、いくつかの共通の構成要件を示し、サーバーを構成および構成解除する方法、サーバーを開始および停止する方法、およびサーバー情報を表示する方法について説明します。

共通のサーバー構成要件

サーバーを構成する前に、いくつかの準備ステップを実行する必要があるかもしれません。サーバーが DCE の認証と許可を使用している場合、そのプリンシパル名を DCE セキュリティー・サービスを使用して登録するか、または親プロセスの DCE 識別のもとで実行しなければなりません。サーバー・アカウントの作成の詳細については、 395ページの『第31章 アカウントの作成および保守』を参照してください。

サーバー構成情報の名前の指定

サーバー構成情報は、***/.../cellname*/hosts/*hostname*/config/srvrconf/*servername* という形式の名前を使用してアクセスすることができます。必要な許可を持っていれば、グローバル名を使用して、リモート・ホスト (別のセル内のホストでもよい) 上の構成データベースにアクセスすることができます。以下の例は、リモート・セル **/.../their_cell.goodco.com** の中のホスト **krypton** 上の、 **video_clip** サーバーに対する構成情報を表示します。

```
dcecp> server show /.../their_cell.goodco.com/hosts/krypton/config/srvrconf/video_clip
{uuid 2fa417e8-bb4c-11cd-831b-0000c08adf56}
{program {vclip}}
{arguments {-catalog}}
 .
 . (Output Omitted)
 .
dcecp>
```

次の例は、ローカル・セル内のホスト **silver** 上の **video_clip** サーバーに対する構成情報を表示します。

```
dcecp> server show /.:/hosts/silver/config/srvrconf/video_clip
{uuid 2fa417e8-bb4c-11cd-831b-0000c08adf56}
{program {vclip}}
{arguments {-catalog}}
 .
 . (Output Omitted)
 .
dcecp>
```

ローカル・ホスト上の **video_clip** サーバーに対する構成情報を表示するには、単純名を使用します。

```
dcecp> server
show video_clip
{uuid 2fa417e8-bb4c-11cd-831b-0000c08adf56}
{program {vclip}}
{arguments {-catalog}}
 .
 . (Output Omitted)
 .
dcecp>
```

サーバー構成情報

各 DCE には、当該ホスト上のサーバーに関する構成情報を格納できるデータベースがあります。 DCE 制御プログラム **server** オブジェクトを使用して、ホスト・システム上のサーバー構成データベース内のサーバー構成情報を格納、修正、または削除することができます。

サーバー構成を管理する場合、次に示す情報の一部または全部を指定する必要があります。

uuid　特定のサーバー構成オブジェクトの識別子。

program
サーバー・プログラムを呼び出す名前 (パス名を含みます)。

directory
プログラムの作業ディレクトリーの名前。サーバーの実行後、出力ファイルまたは一時ファイルを保管する場所が必要になることがあります。

arguments
サーバーを開始するために使用されるコマンド行引き数。

entryname
サーバーがバインディングをエクスポートする先の RPC エントリーの名前。

keytabs
サーバーがそのキーを格納する、関連 keytab オブジェクト (ファイル) の 1 つまたは複数の UUID のリスト。この情報は、DCE の認証または許可を使用するサーバーに必要です。

principals
DCE セキュリティー・サービスに登録される、サーバーの 1 つまたは複数のプリンシパル名のリスト。この情報は、DCE の認証または許可を使用するサーバーに必要です。

services
サーバーが提供するサービスを識別します。各サービス属性は、次の要素が入っている属性リストから構成されます。

annotation
サービスを記述する、人間に読み取り可能なストリング。

ifname
(インターフェース定義ファイルに指定されている) このサービスのインターフェース名。

interface
(インターフェース定義ファイルに指定されている) このサービスのインターフェース*識別子* (UUID およびバージョン番号)。

binding
このサービスを識別するストリング・バインディングのリスト。

entryname
サーバーがこのサービスに関するバインディングをエクスポートする先の、RPC エントリーの名前。

flags このサーバーのフラグを識別するためのキーワードのリスト。**使用不能**フラグだけが、現在サポートされています。

objects
このサービスによってサポートされる、オブジェクト UUID のリスト。

uid サーバーが開始される POSIX UID。

starton
サーバーの開始条件を指定します。値は、次に示す値の 1 つまたは複数からなるリストです。

auto サーバーは、そのサービスに対する要求を DCE デーモンが受け取ると、開始されます。

explicit
サーバーは、そのサーバーの名前を直接指定する **server start** オペレーションまたは **server stop** オペレーションを管理者が実行すると、開始または停止されます。

boot サーバーは、ホスト・システムが開始されると必ず開始されます。

failure サーバーは、異常終了状態で終了したときに、開始されます。

サーバー制御機能をアクセスするための許可

ACL は、サーバー制御機能によって維持されている情報を、無許可のプリンシパルが作成、読み取り、変更、または削除するのを防止します。

サーバー制御機能は、2 種類のサーバー制御情報を維持します。サーバー構成情報 (DCE では **srvrconf** という名前が付いています) は、サーバーを開始するために必要な情報からなります。サーバー実行情報 (DCE では **srvrexec** という名前が付いています) は、実行中のサーバーを制御または停止するために必要な情報からなります。

サーバー構成情報は、2 種類の ACL によって保護されます。 1 つの ACL は、サーバー制御情報が常駐するコンテナーを保護します。もう 1 つの種類の ACL は、各サーバーの構成情報を保護します。

これと同様に、サーバー実行情報も 2 種類の ACL によって保護されます。 1 つの ACL は、サーバー実行情報が常駐するコンテナーを保護します。もう 1 つの種類の ACL は、実行中の各サーバーの実行情報を保護します。

ここでは、サーバー制御情報を保護する ACL の管理方法について説明します。ACL 保護の設定および使用の詳細については、 329ページの『第28章 アクセス制御リストの使用』を参照してください。

サーバー構成コンテナーに関する許可: サーバー構成情報は、コンテナー に置かれています。コンテナーは、UNIX システムでファイルとして使用される補助記憶メカニズムであり、ルートによって所有され、ACL によって保護もされます。このような ACL 許可は、コンテナー内の情報にアクセスできる人を制御します。各 DCE ホストは、以下の名前のサーバー構成コンテナー ACL を 1 つ持っています。

/.../*cellname*/**hosts/***hostname*/**config/srvrconf**

サーバー構成コンテナー ACL は、以下の許可を持っています。

c (制御)
コンテナー ACL を変更する。

r (読み取り)
コンテナー内の構成情報を読み取る。

i (挿入)
新しい構成情報を作成する。

I (挿入)
特権ユーザー (たとえば、POSIX システム上のルート) として実行する、サーバーに関する新しい構成情報を作成する。このオペレーションには、**i** 許可も必要です。

ACL を表示または変更するには、**dcecp acl** オブジェクトを使用します。たとえば、以下のオペレーションを使用して、ホスト **silver** 上のサーバー構成コンテナー・オブジェクトの ACL を表示することができます。

```
dcecp> acl show /.:/hosts/silver/config/srvrconf
{user appl_admin criI}
{unauthenticated r}
{any_other r}
dcecp>
```

/.:/hosts/silver/config/srvrconf はコンテナーであるため、初期コンテナー ACL と初期オブジェクト ACL も持ちます。 **acl** オペレーションで **-ic** オプションおよび **-io** オプションを使用すれば、これらの初期 ACL を操作することができます。ただし、現在は、**/.:/hosts/***hostname***/config/srvrconf** の下に子コンテナーを作成することができないので、初期コンテナー ACL は何の効果も持たないことに注意してください。

サーバー構成情報をアクセスするための許可: 各サーバーの構成情報は、それ自身の ACL によって保護されます。これらの ACL は、無許可のプリンシパルがサーバー構成情報を作成、読み取り、変更、または削除するのを防ぎ、また、サーバーを開始、停止、使用可能化、および使用不能化するのを防ぐことができます。

各 ACL は、それが保護するサーバー構成情報にちなんで名前が付けられ、以下のような名前を持っています。

/.../*cellname*/**hosts/***hostname*/**config/srvrconf/***server_name*,

この ACL は次の許可を持っています。

c (制御)
ACL を変更する。

d (削除)
サーバー構成情報を削除する。

f (フラグ)
カスタム・フラグを使用してサーバーを開始する。

r (読み取り)
サーバー構成情報を読み取る。

w (書き込み)
サーバー構成情報を変更する。

x (実行)
サーバーを開始する。

ACL を表示または変更するには **acl** オブジェクトを使用します。たとえば、以下のオペレーションを使用すれば、ホスト **silver** 上の **video_clip** サーバーの ACL を表示することができます。

```
dcecp> acl show /.:/hosts/silver/config/srvrconf/video_clip
{user appl_admin cdfrwx}
{unauthenticated r}
{any_other r}
dcecp>
```

この ACL は、コンテナーの初期オブジェクト ACL からデフォルト値を取得します。 **acl** オペレーションで **-io** オプションを使用すれば、初期オブジェクト ACL を操作することができます。以下の例は、**video_clip** サーバーの初期オブジェクト ACL を表示します。

```
dcecp> acl show /.:/hosts/silver/config/srvrconf/video_clip
-io
{unauthenticated r}
{any_other r}
dcecp>
```

サーバー実行コンテナーに関する許可: サーバーが開始されると、DCE デーモンは、サーバー構成情報をサーバー実行データベースにコピーします。 **dced** プロセスは、UUID、サーバーの通信エンドポイント、およびそのプロセス名と ID など、実行中のサーバーに関する情報も追加します。実行情報は、実行中のサーバーを制御します。たとえば、プロセス ID はサーバーを停止するために使用されます。サーバーが終了すると、DCE デーモンはそのサーバー実行情報を削除します。

サーバー実行情報は、コンテナー に置かれています。コンテナーは、UNIX システムでファイルとして使用される補助記憶メカニズムであり、これは root によって所有され、**dced** を介するそのアクセスは ACL によって保護されます。このような ACL 許可は、コンテナー内の情報にアクセスできる人を制御します。各 DCE ホストには、以下の名前のサーバー実行コンテナー ACL が 1 つあります。

/.../*cellname*/**hosts**/*hostname*/**config**/**srvrexec**

このサーバー実行コンテナー ACL は、以下の許可を持っています。

c (制御)
コンテナー ACL を変更する。

r (読み取り)
コンテナー内の実行情報を読み取る。

i (挿入)
新しい実行情報を作成する。

I (挿入)
特権ユーザー (たとえば、ルート) として実行する、サーバーに関する新しい実行情報を作成する。このオペレーションには、**i** 許可も必要です。

ACL を表示または変更するには **acl** オブジェクトを使用します。たとえば、以下のオペレーションを使用して、ホスト **silver** 上のサーバー実行コンテナー・オブジェクトの ACL を表示することができます。

```
dcecp> acl show /.:/hosts/silver/config/srvrexec
{user appl_admin criI}
{unauthenticated r}
{any_other r}
dcecp>
```

/.:/hosts/silver/config/srvrexec はコンテナーであるため、初期コンテナー ACL および初期オブジェクト ACL も持っています。 **acl** オペレーションで **-ic** オプションおよび **-io** オプションを使用すれば、これらの初期 ACL を操作することができます。現在、**/.:/hosts/***hostname***/config/srvrexec** の下に子コンテナーが存在していないので、初期コンテナー ACL は何の効果も持たないことに注意してください。

サーバー実行情報をアクセスするための許可: 各サーバーの構成情報は、それ自身の ACL によって保護されます。これらの ACL は、無許可のプリンシパルが、サーバー構成情報を作成、変更、読み取り、または削除するのを防ぎ、また、サーバーを開始、停止、使用可能化、および使用不能化するのを防ぐことができます。

各 ACL は、それが保護するサーバー実行情報にちなんで名前が付けられ、以下のような名前を持っています。

/.../*cellname*/**hosts**/*hostname*/**config**/**srvrexec**/*server_name*

この ACL は次の許可を持っています。

c (制御)
ACL を変更する。

r (読み取り)
サーバー実行情報を読み取る。

w (書き込み)
サーバー実行情報を変更する。

s (停止)
サーバーを停止する。

たとえば、以下のオペレーションを使用すれば、ホスト **silver** 上の **video_clip** サーバーのサーバー実行情報の ACL を表示することができます。

```
dcecp> acl show /.:/hosts/silver/config/srvrexec/video_clip
{user appl_admin crws}
{unauthenticated r}
{any_other r}
dcecp>
```

この ACL は、コンテナーの初期オブジェクト ACL からデフォルト値を取得します。 **acl** オペレーションで **-io** オプションを使用すれば、初期オブジェクト ACL を操作することができます。以下の例は、**video_clip** サーバーの初期オブジェクト ACL を表示します。

```
dcecp> acl show /.:/hosts/silver/config/srvrexec/video_clip
-io
{unauthenticated r}
{any_other r}
dcecp>
```

サーバーの構成

server create オペレーションを使用して、アプリケーション・サーバーがサーバー制御機能にアクセスできるようにします。サーバーの構成は、サーバーの開始または制御に必要な情報の作成を意味します。一般に、これには、サーバーの開始コマンド行と引き数、および DCE アプリケーションを開始するために必要なその他の情報が含まれます。

ホスト・システムが実行中の場合、いくつかのサーバーが使用可能でなければなりません。たとえば、ホスト活動に関する情報を提供するサーバーを、ホストのブート時に開始し、ホストがシャットダウンするまで実行させたい場合があります。他の種類のサービスがほんの短期間だけ必要な場合もあります。サーバー制御機能には、サーバーを開始および停止するためのいくつかの条件を指定できるようにするための、管理インターフェースがあります。

- **explicit**: 希望するときにいつでも明示して開始できるように、サーバーを設定することができます。
- **boot**: ブート時に開始するようにサーバーを設定することができます。
- **automatic**: 要求された時点でサーバーを開始するよう設定することができます。すなわち、クライアントからのサービス要求がホスト・システムで受け取られた時に開始します。
- **failure**: サーバーが予期せずに終了した場合に自動的に開始するように、サーバーを設定することができます。

以下の例は、ローカル・ホスト上で **video_clip** という名前の架空のビデオ・クリップ・サーバーに関するエントリーを作成します。リモート・ホストまたは別のセル内のホストの場合、セル相対名またはグローバル名を使用してください。 **vclip** という名前のプログラムは、 **/usr/local/bin** 作業ディレクトリー内にあるサーバーを起動します。このサーバーは、引き数として **-catalog** を指定することによって設定された、カタログ・モードになっています。サーバーは DCE セキュリティー・サービスを使用するので、このサーバーのプリンシパル名は **Vclip_Srv_1** になります。 **-entryname** オプションは、サーバーがバインディング情報を保管する、セル・ディレクトリー・サービス (CDS) 内のエントリー名を指定します。**-starton** オプションは、**video_clip** サーバーを指定した明示的な **server start** オペレーションを **dced** が受け取った時に開始するよう、サーバーを設定します。**failure** 属性によって、さらに、サーバーが異常な状態で終了した場合に、サーバー

を再起動することが指定されています。 **-services** オプションには注釈情報があるので、 **server show** オペレーションによってこの情報が戻された時に、管理者はサーバーを識別することができます。 **interface** 属性は必要です。その理由は、サーバーの開始時に、 DCE デーモンがこの情報をホストのエンドポイント・マップにコピーするためです。

```
dcecp> server create /.:/hosts/silver/config/srvrconf/video_clip
¥
> -program {/usr/local/bin/vclip} ¥
> -directory {/tmp} -arguments {-catalog} ¥
> -principal {Vclip_Srv_1} ¥
> -entryname {/.:/subsys/applications/video_clip_1} ¥
> -starton {explicit failure} ¥
> -services {{annotation {Video Clip Catalog and Server}} ¥
> {interface {d860322b-d499-11cd-9dfb-0000c08adf56 1.0}}}
dcecp>
```

次の例は、ホスト・システムのブート時に開始するように、同じサーバーを構成しています。上記の例との唯一の違いは、 **-starton** オプションの値が **boot** である点です。

```
dcecp> server
create /.:/hosts/silver/config/srvrconf/video_clip ¥
> -program {/usr/local/bin/vclip} ¥
> -directory {/tmp} -arguments {-catalog} ¥
> -principal {Vclip_Srv_1} ¥
> -entryname {/.:/subsys/applications/video_clip_1} ¥
> -starton {boot} ¥
> -services {{annotation {Video Clip Catalog and Server}} ¥
> {interface {d860322b-d499-11cd-9dfb-0000c08adf56 1.0}}}
dcecp>
```

最後の構成例では、サービスに対するクライアント要求がホスト・システムで受信されたときに、必ず **video_clip** サーバーを開始するように設定しています。 **-starton** オプションの値は **auto** です。 126ページの『サーバーの構成解除』で、サービスを使用不能および使用可能にするステップについて説明します。

```
dcecp> server create /.:/hosts/silver/config/srvrconf/video_clip ¥
> -program {/usr/local/bin/vclip} ¥
> -directory {/tmp} -arguments {-catalog} ¥
> -principal {Vclip_Srv_1} ¥
> -entryname {/.:/subsys/applications/video_clip_1} ¥
> -starton {auto} ¥
> -services {{annotation {Video Clip Catalog and Server}} ¥
> {interface {d860322b-d499-11cd-9dfb-0000c08adf56 1.0}}}
dcecp>
```

サーバー構成情報のリストおよび検索

特定のホスト上に構成されているサーバーの名前のリストを表示するには、以下に示されるように、**server catalog** オペレーションを使用してください。このオペレーションは、ホスト上で使用可能なすべてのサーバーを表示するのではなく、構成情報がサーバー構成データベースに格納されているサーバーだけを表示します。

```
dcecp> server
catalog /.:/hosts/silver
/.../my_cell.goodco.com/hosts/silver/config/srvrconf/video_clip
dcecp>
```

セル内の各ホストについて **server catalog** オペレーションを繰り返すには、**foreach** コマンドを使用して、DCE セル内のすべての構成されたサーバーの名前をリストします。

```
foreach h [directory list /.:/hosts]{
   echo [server catalog $h]
}
```

あるサーバーについて確立された構成情報がよくわからない場合は、以下に示すように、**server show** オペレーションを使用して、その構成情報を表示させることができます。実行中のサーバーに関する情報を表示するには、**-executing** オプションを使用してください。

```
dcecp> server show /.:/hosts/silver/config/srvrconf/video_clip
{uuid d860322b-d499-11cd-9dfb-0000c08adf56 1.0}
{program {/usr/local/bin/vclip}}
{arguments {-catalog}}
{prerequisites {}}
{keytabs {683cf29a-e456-11cd-8f04-0000c08adf56}}
{services {{annotation "Video Clip Catalog and Server"}}
{principals {Vclip_Srv_1}}
{starton {explicit failure}}
{uid 1441}
{gid 1000}
{dir {/tmp}}
dcecp>
```

サーバーの構成解除

server delete オペレーションを使用すれば、ホストの構成データベースからサーバー構成情報を削除することができます。このオペレーションは、たとえば、サーバーが別のホストに移動する場合に実行します。 **server delete** オペレーションは、現在実行中のサーバーを停止しません。

以下の例は、ホスト **silver** 上の構成データベースから **video_clip** サーバーの構成情報を削除します。

```
dcecp> server delete /.:/hosts/silver/config/srvrconf/video_clip
dcecp>
```

サーバーの開始および停止

サーバーが適切に構成されれば、**server start** または **server stop** オペレーションを使用して、サーバーをリモートで開始または停止することができます。たとえば、以下の **server start** オペレーションは、ローカル・セル内のホスト **silver** 上でサーバー **video_clip** を開始します。

```
dcecp> server start /.:/hosts/silver/config/srvrconf/video_clip
eb814e2a-0099-11ca-8678-02608c2ea96e
dcecp>
```

次の例は、ローカル・セル内のローカル・ホスト **silver** 上のサーバー **video_clip** を停止します。

```
dcecp> server stop video_clip
dcecp>
```

サービスの使用不能化および使用可能化

サーバーが実行中であっても、そのサービスを使用不能として設定することによって、サーバーが提供するサービスをクライアントに使用させないようにすることができます。使用不能として設定すると、サーバーのエンドポイント情報が要求側のクライアントに戻されないので、クライアントはサーバーを検出することができなくなります。その代わりに、クライアントは、 エンドポイントが登録されていないというサーバー状況を受け取ります。ただし、すでにサーバー・エンドポイントを獲得しているクライアントは、サーバーとの通信を行うことができます。

サーバーに複数のインターフェースがある場合、インターフェース識別子を指定することによって、1 つまたは複数のインターフェースを使用不能化することができます。以下の例は、 **video_clip** サーバーの 1 つのサービスを使用不能にします。

```
dcecp> server disable /.:/hosts/silver/config/srvrexec/video_clip ¥
> -interface {d860322b-d499-11cd-9dfb-0000c08adf56 1.0}
dcecp>
```

次の例は、**video_clip** サーバーの **vidsrv** サービスを使用不能にした後で、それを使用可能にします。このオペレーションによって、クライアントはサーバーのエンドポイントを獲得できるようになります。

```
dcecp> server enable /.:/hosts/silver/config/srvrexec/video_clip
¥
> -interface {d860322b-d499-11cd-9dfb-0000c08adf56 1.0}
dcecp>
```

サーバー構成の拡張

一部のサーバーは、DCE ソフトウェアが提供する属性のセットによってサポートされない、構成情報を必要とする場合があります。 **xattrschema** オブジェクトを使用して追加の*拡張レジストリー属性* (ERA) を作成することによって、任意の情報をサーバー構成情報に追加することができます。

たとえば、オブジェクト・ファミリーを指定する属性が必要なサーバーがあるとします。そのような属性は、**xattrschema** オブジェクトを使用して作成します。以下の例は、**srvrconf/objfamily** という ERA を作成します。このオペレーションは、ERA の照会、更新、テスト、および削除に必要な許可を指定し、また、このような許可をサポートする ACL マネージャーを指定します。

```
dcecp> xattrschema create ¥
> /.:/hosts/silver/config/xattrschema/srvrconf/objfamily ¥
> -attribute {{annotation {object family}} {encoding uuid} ¥
> {aclmgr {srvrconf r w r d}}}
dcecp>
```

新しい属性を作成したら、129ページの『サーバー構成の変更』で説明したように、**server modify** オペレーションを使用して、必要なデータを挿入します。 ERA についての詳細は、417ページの『第32章 拡張レジストリー属性の作成および使用』にあります。

以下の例に示すように、**xattrschema show** オペレーションを使用して、 ERA に関連する属性を調べることができます。

```
dcecp> xattrschema show /.:/hosts/silver/config/xattrschema/srvrconf/objfamily
{aclmgr {srvrconf {{query r} {update w} {test r} {delete d}}}}
{annotation {object family}}
{applydefs no}
```

```
{encoding uuid}
{intercell reject}
{multivalued yes}
{reserved no}
{scope {}}
{trigbind {}}
{trigtype none}
{unique no}
{uuid 1bef2222-e687-11cd-b74a-0000c08adf56}
dcecp>
```

サーバー構成情報内の ERA は、2 つのレベルの ACL によって保護されます。 1 種類の ACL は、ERA が常駐しているコンテナーを保護します。もう 1 種類の ACL は、各 ERA を保護します。

ERA コンテナー ACL は、以下のような名前になっています。

/.../*cellname*/**hosts/***hostname*/**config/xattrschema**

ERA コンテナー ACL は、以下の許可を持っています。

c (制御)
: コンテナー ACL を変更する。

r (読み取り)
: コンテナー内の ERA を読み取る。

i (挿入)
: 新しい ERA 情報を作成する。

l (挿入)
: **l** 許可は存在していますが、ERA 項目には適用されません。この許可はサーバー制御機能に適用されますが、これについては、120ページの『サーバー制御機能をアクセスするための許可』で説明します。

コンテナー ACL を表示または変更するには、**dcecp acl** オブジェクトを使用します。たとえば、以下のオペレーションは、ホスト **silver** 上の ERA コンテナー ACL を表示します。

```
dcecp> acl show /.:/hosts/silver/config/xattrschema
{user appl_admin criI}
{unauthenticated r}
{any_other r}
dcecp>
```

個々の ERA の ACL は、以下のような名前になっています。

/.../*cellname*/**hosts/***hostname*/**config/xattrschema/***ERA_name*

各 ERA 上の ACL は、無許可のプリンシパルによる ERA 情報の作成、読み取り、変更または削除を防ぐことができます。以下の例は、**srvrconf/objfamily** ERA に対して確立された許可を表示します。この例では、**c** 許可は、ERA が **xattrschema create** オペレーションによって作成された時に割り当てられていないので、何の効果もありません。すべてのユーザーが ERA について照会し、テストすることができます。 **appl_admin** という名前のユーザーだけが、ERA の更新および削除もできます。

```
dcecp> acl show /.:/hosts/silver/config/xattrschema/srvrconf/objfamily
{user appl_admin crwd}
{unauthenticated cr}
{any_other cr}
dcecp>
```

この ACL は、コンテナーの初期オブジェクト ACL からデフォルト値を取得します。 **acl** オペレーションで **-io** オプションを使用すれば、初期オブジェクト ACL を操作することができます。以下の例は、ホスト **silver** 上の **xattrschema** コンテナーに関する初期オブジェクト ACL を表示します。

```
dcecp> acl show /.:/hosts/silver/config/xattrschema -io
{unauthenticated cr}
{any_other cr}
dcecp>
```

サーバー構成の変更

サーバーの構成情報を変更したい場合があります。たとえば、**-starton** 属性を **boot** から **explicit** に変更して、サーバーを手動で制御したい場合があります。

通常のサーバー構成属性を変更するには、まず最初に既存のすべての属性を削除してから、新しい属性を作成する必要があります。現在の情報を失わないようにするために、最初に **server show** オペレーションを使用して、その情報を画面に表示させてください。

そのステップが以下の例で示されています。ここでは現行のサーバー構成情報を入手するために **server show** オペレーションを使用しています。 **server delete** オペレーションは構成情報を削除し、**server create** オペレーションは、新しい **-starton** 属性を残りのサーバー構成情報とともに挿入します。

```
dcecp> server show /.:/hosts/silver/config/srvrconf/video_clip
{uuid d860322b-d499-11cd-9dfb-0000c08adf56 1.0}
{program {/usr/local/bin/vclip}}
{arguments {-catalog}}
{prerequisites {}}
{keytabs {683cf29a-e456-11cd-8f04-0000c08adf56}}
{services {{annotation "Video Clip Catalog and Server"}}
{principals {Vclip_Srv_1}}
{starton {boot}}
{uid 1441}
{gid 1000}
{dir {/tmp}}
dcecp> server delete /.:/hosts/silver/config/srvrconf/video_clip
dcecp> server create /.:/hosts/silver/config/srvrconf/video_clip ¥
> -program /usr/local/bin/vclip ¥
> -directory /tmp ¥
> -arguments {-catalog} ¥
> -principal Vclip_Srv_1 ¥
> -entryname /.:/subsys/applications/video_clip_1 ¥
> -starton {explicit} ¥
> -services {{annotation "Video Clip Catalog and Server"}}
dcecp>
```

server modify オペレーションを使用して、 ERA 情報を直接変更することができます。以下の例は、 **srvrconf/objfamily** というサーバーの ERA を新しい値を含むよう変更します。このオペレーションでは、 127ページの『サーバー構成の拡張』で説明した **xattrschema create** オペレーションを使用して、 ERA がすでに作成されているものと想定しています。

```
dcecp> server modify /.:/hosts/silver/config/srvrconf/video_clip
¥
> -change {srvrconf/objfamily {c09dcc40-e4f4-11cd-bd59-0000c08adf56}}
dcecp>
```

サーバーが実行中であるかどうかの検査

server ping オペレーションを実行することによって、特定のサーバーが実行中であるかどうかをチェックすることができます。クライアント・ユーザーがサーバーと通信できないと報告してきた場合に、これは便利なテスト方法です。**server ping** オペレーションは、指定されたサーバーと通信して、サーバーが存在しているかどうかをテストし、サーバーが listen していれば **1** を戻し、サーバーが listen していなければ **0** を戻します。 server ping オペレーションの引き数は、**srvrconf** オブジェクトの名前ではなく、サーバーの **entryname** です。以下の例は、**video_clip** サーバーが実行中であるかどうかをテストします。

```
dcecp> server ping /.:/subsys/applications/video_clip_1
1
dcecp>
```

クライアント / サーバーのバインディング情報の管理

DCE 環境では、クライアントとそのサーバーがネットワーク内の異なるホスト上に常駐している場合が多いため、クライアントにはサーバーを見つける方法が必要です。

クライアントがサーバーと通信するには、次の 3 つの情報が必要です。

- サーバーが実行中のホストのホスト名 (またはネットワーク・アドレス)
- サーバーが使用中のネットワーク・トランスポートの名前
- サーバーがクライアント通信で使用している通信ポート (エンドポイント)

もちろん、アプリケーション・プログラマーは、サーバーの所在情報 (*バインディング情報* とも呼ばれます) を、アプリケーションのクライアント側 (その情報が即時に使用できるところ) に単にハードコーディングすることができます。しかし、この方法の場合は、プログラマーは正確なネットワーク詳細 (ホスト名や使用可能なポート番号など) に関する事前の知識を持っていなければなりません。その上、ハードコーディングされたバインディング情報を持ったサーバーは、構成変更に容易に対応することができません。サーバーを別のホストに移動した場合、サーバーの新しいホスト名を使用して、すべてのクライアントを再コンパイルする必要があります。そこで DCE には、クライアントがサーバー・バインディングを取得するための、より柔軟性のある方法が用意されています。

クライアントがサーバーを検出する標準の方法では、 CDS およびサーバー・ホストのエンドポイント・マップを使用します。 131ページの図1 は、ホスト **larry** 上の架空の辞書クライアント・アプリケーションが、ホスト **curly** 上の辞書サーバーを検出する方法について、高いレベルから見た例を示しています。

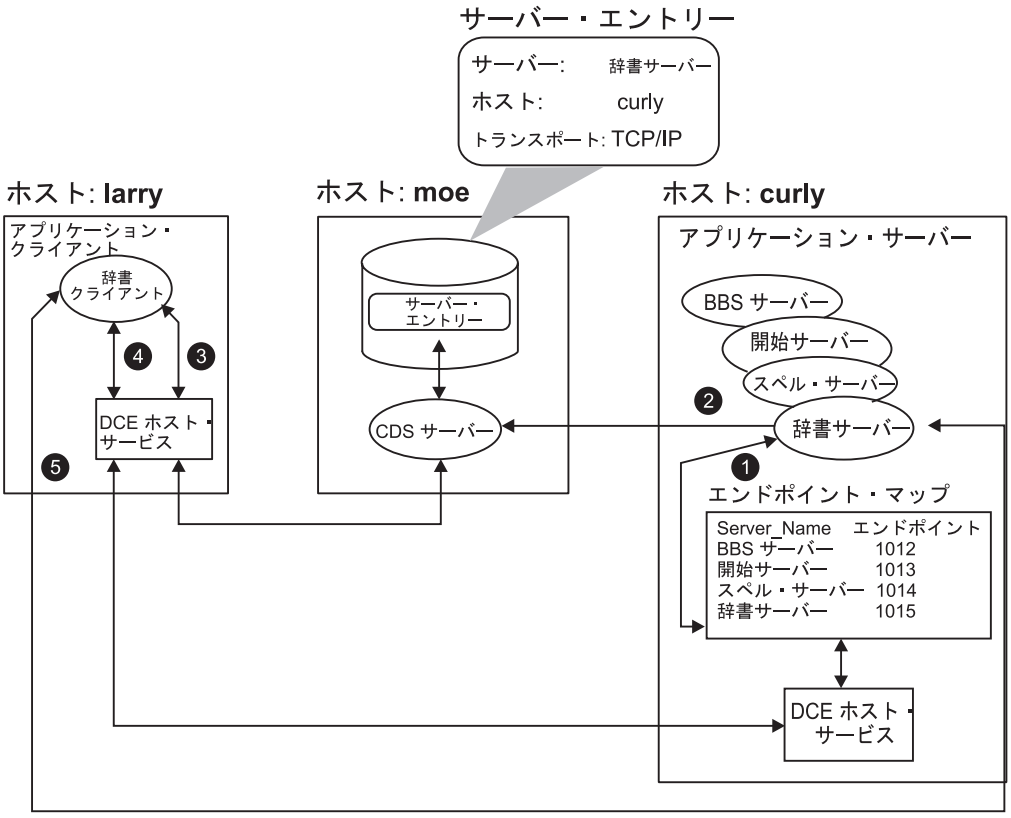


図 1. サーバー・バインディング情報

1. 辞書サーバーが起動すると、DCE ホスト・ソフトウェアはこのサーバーに通信ポート (エンドポイント) を割り当てます。クライアントはこの通信ポートを使用して、このサーバーと通信します。ここで、エンドポイントは TCP/IP ポート 1015 です。 DCE ホスト・ソフトウェアは、サーバー識別情報と現在のエンドポイントも、ホストのエンドポイント・マップに置きます。
2. 次に、辞書サーバーは、そのホスト名 (通常はホスト・アドレス) および使用するトランスポートを CDS 内のサーバー・エントリーに入れて (エクスポート して)、その使用可能性をクライアントに通告します。
3. 辞書クライアントが、サーバーによって提供されるリモート・プロシージャーを呼び出すと、クライアント上の DCE ソフトウェアが、CDS サーバーに照会して、辞書サーバーのホスト名およびトランスポートを検出します。
4. 次に、クライアント・システムのホスト・ソフトウェアが、ホスト **curly** 上のエンドポイント・マップに照会して、辞書サーバーのエンドポイント (ポート 1015) を検出します。
5. 必要なすべてのバインディング情報が揃うと、ホスト **larry** 上のホスト・サービスは、リモート・プロシージャー・コールをホスト **curly** 上のポート 1015 に直接伝送します。

この高いレベルから見た例では一部の詳細が省略されていますが、この図は、クライアントとサーバーが行う主要なバインディング活動を示しています。すなわち、サーバーは、CDS とホスト・エンドポイント・マップ内に、クライアントが探索するバインディング情報を置きます。クライアントがサーバーを検出する他の方法もあります。また、上記のメカニズム自体にもさまざまなバリエーションがありま

す。しかし、こうした代替方法は、通常、**dcecp** のような従来の DCE 管理機能を介してではなく、アプリケーション自体によって制御されます。

この項では、1 つの基本的なクライアント / サーバー・バインディング・メカニズムについて説明します。次の項では、エンドポイント・マップおよび CDS によって実行される役割について調べます。さらに、エンドポイント・マップおよび CDS 内のバインディング情報を管理するための特定の管理タスクについても説明します。

アプリケーションの容易な開発と管理のためのエンドポイント・マップの使用

リモート・クライアントは、サーバー・ホストのエンドポイント・マップを使用して、サーバーの通信エンドポイントを判別することによって、サーバーを検出できます。しかし、リモート・クライアントは、どのようにしてエンドポイント・マップ自体を検出する場所を知るのでしょうか。リモート・クライアントは、エンドポイント・マップが各ホスト上のウェルノウン・エンドポイント (すなわち、常に同じエンドポイント) で常にアクセス可能であるため、それを知っており、これらのクライアントは容易に検出することができます。

ホストが複数のトランスポートをサポートしている場合、エンドポイント・マップは、各トランスポートごとに 1 つのポート上で*listen* します。 IP アドレス・ファミリー (TCP と UDP の両方) 内では、エンドポイント・マップ・プロセスはポート 135 上で listen します。ドメイン・ドメイン・ソケット (DDS) アドレス・ファミリーでは、ポート 12 上で listen します。 DECnet NSP アドレス・ファミリーでは、ポート 69 上で listen します。エンドポイント・マッパー・サービスによって使用される、プロトコル・シーケンスおよびウェルノウン・エンドポイント・マップの完全なリストは、ヘッダー・ファイル **/opt/dcelocal/share/include/dce/ep.idl** の中に入っています。

すべてのホストがすべてのトランスポートをサポートするわけではないことに注意してください。 DCE ソフトウェアは、クライアントとサーバーの間で少なくとも 1 つのトランスポートが共用されていることを確認しようとします。

ウェルノウン・エンドポイントは重要なサーバーへの便利なアクセスを提供しますが、大部分のサーバーにとって、このようなエンドポイントは実用的ではありません。なぜなら、アドレス・ファミリーによっては、エンドポイントの数が限定されていて、ウェルノウン・エンドポイントは、中央管理機関によってのみ割り当てられるからです。したがって、ほとんどのサーバーは、*動的エンドポイント* を使用します。サーバーが起動すると、RPC 実行時ライブラリーが、オペレーティング・システムから使用可能なエンドポイントを入手し、ホストのエンドポイント・マップに登録します。

サーバーが開始されるごとに異なるエンドポイントがサーバーに割り当てられることがあるため、エンドポイント情報は、 CDS ではなく、エンドポイント・マップに保管されます。なお、CDS は、サーバーのホスト・アドレスおよびサーバーが使用するトランスポートといった、より安定性の高い情報のためのリポジトリーです。サーバーが同じマシンにとどまる限り、ホスト情報とトランスポート情報は更新する必要がありません。これにより、CDS でのボトルネックが削減されます。

この構造によって、エンドポイントを管理する必要がなくなるので、アプリケーションの開発と管理がより容易になります。サーバー側では、動的エンドポイントをクライアントに渡すことを気にしなくて済みます。さらに、サーバーが新しいホストに移動しない限り、あるいはトランスポートを削除したり追加したりしない限り、CDS 内の情報を更新する必要もなくなります。

エンドポイント・マップの自動管理

エンドポイント・マップを使用する各サーバーは、起動時に 1 組の情報をエンドポイント・マップに格納します。この情報には、オブジェクトのユニバーサル・ユニーク識別子 (UUID) とサーバーが提供するインターフェース、注釈ストリング、およびその他のフィールドが含まれます。

エンドポイント・マップは、ディスクの *dcelocal***/var/dced/Ep.db** および *dcelocal***/var/dced/Srvrexec.db** の中に常駐します。システムのリブート後、DCE ベースのサーバーは、再起動してエンドポイント・マッパー・サービスに再登録するので、 DCE デーモンが起動する前にデータベース・ファイルを削除する必要があります。これは、大部分のシステムで自動的に行われます。

DCE ベースのサーバーは、通常、起動時にエンドポイント・マッパー・サービスに登録し、終了時に登録を抹消する必要があります。サーバーが抹消せずに終了した場合、エンドポイント・マップには古いエントリーが含まれている可能性があります。

DCE には、サーバーを抹消して、エンドポイント・マップ内に古いエントリーを残さないようにするサーバー制御機能が用意されています。このような機能を使用しないサーバー (たとえば、古いサーバー) は、突然に終了した場合に古いエントリーを残しがちです。したがって、定期的に DCE デーモン (**dced**) が、エンドポイント・マップを走査し、登録されている各サーバーを ping し、応答しないサーバーに関するエントリーを削除することによって、古いエントリーをパージします。

古いエントリーを削除する背景プロセスは、高い応答性を意図していません。 RPC をサービスしなくなったサーバーをエンドポイント・マップから抹消する必要性をなくそうということではありません。そうではなくて、この処理は、サーバーの失敗後のクリーンアップのみを意図しています。

ping とパージのメカニズムの動作は実装に依存しますが、通常の実装では、データベースが走査される (すなわち、サーバーが ping され、古いエントリーが削除される) のは、頻繁には行われません (たとえば、1 時間に数回です)。サーバーへの ping が失敗すると、サーバーは、これより短い間隔 (たとえば、5 分ごと) で数回 ping されます。サーバーがそれでも応答しないと、**dced** は、そのエントリーが古いと判断し、データベースから削除します。最終的には、古いサーバー・エントリーが検出されパージされる速度は、データベース内の古いエントリー数に応じて異なり、古いエントリーが多いほど、その検出とパージにかかる時間が長くなります。

エンドポイントの制限

DCE サーバーとクライアントに対するエンドポイント (ポート) の割り当てを、特定のセットに制限することができます。これは、特定のエンドポイントを使用する

ように設計されている DCE 以外のアプリケーションが存在する環境において、DCE サーバーまたはクライアントがそれらのエンドポイントを独占することを心配したくない場合に便利です。

この機能は、**RPC_RESTRICTED_PORTS** 環境変数 (動的割り当てを、クライアントまたはサーバーのアプリケーションを起動する前に制限する必要があるエンドポイントのリストを持ちます) を設定することによって活動化します。**RPC_RESTRICTED_PORTS** は、RPC ランタイムによるサーバー・ポートの動的割り当てのみを制御します。ウェルノウン・エンドポイントには影響を与えません。

次の例は、 5000 から 5110 までと 5500 から 5521 までの範囲の TCP/IP エンドポイントをサーバーが使用するように制限しています。また、UDP/IP エンドポイントを 6500 から 7000 の範囲に制限しています。

```
% set RPC_RESTRICTED_PORTS
¥
        ncacn_ip_tcp[5000-5110,5500-5521]:ncadg_ip_udp[6500-7000]
%
```

CDS のような DCE サーバーに対して **RPC_RESTRICTED_PORTS** を使用するには、セルを起動するたびに、その前に環境変数を設定してください。

この機能は RPC にセキュリティーを追加しないこと、またこの機能はセキュリティーを目的としていないことに注意してください。この機能は、DCE サーバーへの着呼を可能にするために、単にネットワーク・ファイアウォールの構成を容易にするだけです。

エンドポイント・マップ内の情報の表示

各ホスト上のエンドポイント・マップは、必要に応じて古い項目をパージし、ホストの再ブートごとにエンドポイント情報を削除して、ほとんどの部分について自分自身で管理します。したがって、エンドポイント・マップの管理は、実際には必要ありません。

しかし、クライアント / サーバー通信問題が生じた場合、エンドポイント・マップに格納されている情報が (特にサーバーがクライアントに正しいエンドポイント情報を提供しているかどうかを判定するために)、管理者の役に立つことがあります。この場合、**endpoint** オブジェクトを使用して、エンドポイント・マップ情報を表示することができます。 **endpoint** オブジェクトは、トラブルシューティングで使用するだけでなく、既存のマッピングへの新しいオブジェクト UUID の追加など、他の特別なサーバー・オペレーションのためにも使用することができます。

エンドポイントは ACL によって保護されません。このことは、**dcecp** を実行できる人は誰でも、各自のホスト上で **endpoint show** オペレーションを使用して、セル内の他のホストに関するエンドポイント情報を表示することができることを意味します。エンドポイントの作成または削除のような、その他のエンドポイント・オペレーションは、ローカル・ホストにログインしているユーザーのみが実行することができます。システム管理者特権または root 権限といった、他の特殊な特権は、エンドポイント情報へのローカル・アクセスには必要ありません。

endpoint show オペレーションを使用して、ホストのエンドポイント・マップ・データベースに格納されている情報を表示することができます。以下の例は、リモート・ホスト **megazoid** 上の **video_clip** サーバーについてのエンドポイント・マップ情報を表示します。ローカルのエンドポイント・マップを操作する場合は、*hostname* 引き数を省略してください。

```
dcecp> endpoint show /.:/hosts/megazoid
¥
> -interface {2fa417e8-bb4c-11cd-831b-0000c08adf56 1.0}
{{object 99ff4fb8-c042-11cd-91cd-0000c08adf56}
 {interface {2fa417e8-bb4c-11cd-831b-0000c08adf56 1.0}}
 {binding {ncacn_ip_tcp 130.105.1.227 1028}}
 {annotation {Text Development Utilities}}}
dcecp>
```

オプションを指定せずに **endpoint show** オペレーションを使用して、エンドポイント・マップ内のすべてのエンドポイントを表示することができます。

CDS 内のサーバー・エントリー、グループ、およびプロファイルの管理

エンドポイント・マップは、ホスト上のサーバーのディレクトリーとして機能します。同様に、CDS はセル内のサーバーのディレクトリーとして機能します。この章の最初の部分では、サーバーの名前、そのホスト・アドレス、およびサーバーを使用可能にするトランスポートなど、比較的安定したバインディング情報を保管するためにアプリケーションが CDS を使用する方法について、高いレベルから見た説明を行います。この項では、サーバーおよび他の分散オブジェクトを適切に編成するために CDS 機能を使用する方法を説明します。

以下の項で説明するオペレーションの多くは、 ACL によって無許可アクセスから保護されている、CDS ディレクトリーで操作されます。 ACL および CDS の詳細については、195ページの『第16章 CDS 名へのアクセスの制御』を参照してください。

個々のサーバーおよびオブジェクトを識別するための固有のサーバー・エントリー名の使用

サーバーは、そのバインディング情報を、クライアントが検出できる CDS 内に格納することはすでに述べました。しかし、これまでは、CDS をブラック・ボックスのように扱ってきました。 DCE セルが少数のサーバーまたはオブジェクトとユーザーからなる場合、CDS は、サーバーとクライアントの両方にアクセス可能なデータ・ファイルと同じように単純になります。オブジェクトに固有な名前を見つけることは、大きな問題にはならないはずです。さらに、無許可の使用からオブジェクトを保護するための、効果的な構造を作成することも可能です。しかし、DCE セルには数百、数千というオブジェクトが入る可能性があります。大きなセルには多くの類似した、またはまったく同じのサーバーが入る可能性があるので、そのサービスをクライアントに提供するための便利で効果的な方法が必要になります。

DCE CDS には、この要求に応えるために、階層的な (3 構造の) 名前システムが用意されています。サーバーは、このシステムを使用してバインディング情報を格納します。 CDS は、ディレクトリーの階層ファイル・システムに似ていますが、フ

ァイルの代わりに、名前およびその他の情報を格納します。この階層構造を構築し、会社のオーガニゼーション構造に対応したディレクトリー名を与えることができます。

サーバーには、**/.:/admin/finance/payroll/check_writer** のような CDS 名があります。この **check_writer** サーバーがそのサーバー・エントリー名を CDS にエクスポートすると、 CDS はそれを **/.:/admin/finance/payroll** という名前のディレクトリーに格納します。したがって、クライアントがこの **check_writer** を、**/.:/admin/finance/accts_payable/check_writer** という名前の別の **check_writer** と混同することはありません。このようして、固有のサーバー・エントリー名は、各サーバーにアクセスし制御する方法を提供し、厳しい管理要件を満たすことができます。

本書の 153ページの『第4部 セル・ディレクトリー・サービス』には、CDS および CDS 名の構造と使用法についての詳細が記載されています。ここでは、CDS ディレクトリー名が、機能的に同一のサーバー・エントリーを固有に識別する方法と理由を知るだけで十分です。

サーバー自身がその名前とバインディング情報の CDS からのエクスポートと削除を管理することが多いのですが、管理者が手動でバインディング情報の追加、変更、または削除を行わなければならない場合もあります。たとえば、サーバー・ホスト・マシンが突然クラッシュし、長期にわたってオフラインのままである場合、その常駐サーバーは、そのエントリー名およびバインディング情報を CDS から削除することができません。クライアントがこのような使用できないサーバーを探索して、時間を浪費する可能性があります。 DCE 制御プログラムは **rpcentry** オブジェクトを提供しており、これを使用して CDS 内のサーバー・エントリー名およびバインディング情報を管理することができます。

実際の管理タスクに進む前に、管理する対象のサーバー・エントリーについて調べてみることにします。 137ページの図2 は、サーバー・エントリー内の可能な情報を示したものです。

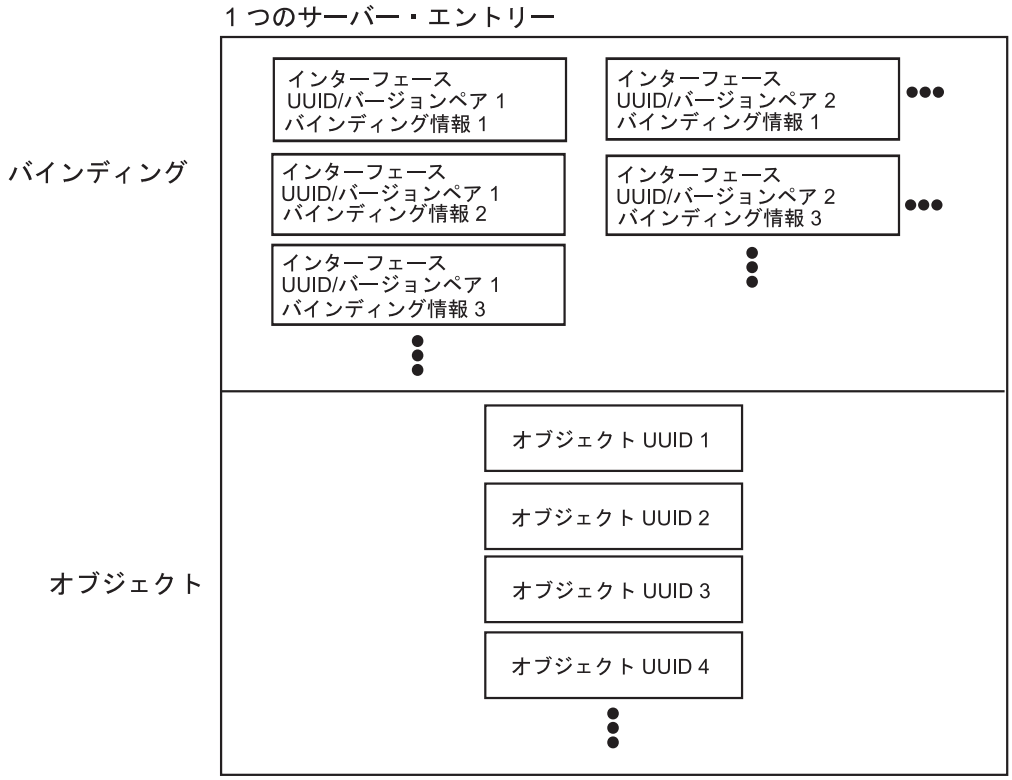


図 2. サーバー・エントリー内の可能な情報

図2 の上部には、バインディングが含まれています。各バインディングは、インターフェース識別子とバインディング情報からなります。インターフェース識別子は、サーバーによって提供されるインターフェースを識別し、一方、そのバインディング情報は、そのインターフェースへのアクセスに使用するホスト・アドレスおよびネットワーク・トランスポートを指示します。以下のバインディングの例 (**dcecp** 構文で示されています) は、サーバーがインターネット・アドレス 120.101.13.157 のホスト上にあり、ユーザー・データグラム・プロトコル (UDP) を使用して使用可能になることを示しています。

```
{nacdg_ip_udg 120.101.13.157}
```

インターフェース識別子が複数のトランスポートで使用可能な場合、サーバー・エントリーにはバインディング (1 つのトランスポートについて 1 つのバインディング) が含まれます。サーバーは複数のインターフェースを提供することができます。複数のインターフェースを、 1 つのエンドポイントを介して使用することができます。すなわち、異なるインターフェースが同じバインディングを持つことができます。

この図の下の部分にはオブジェクト UUID があります。オブジェクト UUID は、サーバーが管理する特定のオブジェクトまたはリソースを識別する追加情報をクライアントに提供します。たとえば、1 つの印刷サーバーは 2 階にあるプリンターを提供し、別の印刷サーバーは 1 階にあるプリンターを提供します。この場合、オブジェクト UUID によって、クライアントは適切な階にあるプリンターを選択することができます。言い換えると、オブジェクト UUID によって、これ以外はまったく同じサービスをクライアントが区別することが可能となります。

アプリケーション・サーバーは CDS 内の自身のサーバー・エントリーを管理することができますが、サーバー・エントリー内の情報を手動で追加、削除、または変更する方がより便利 (かつ単純) な場合もあります。 CDS 内のサーバー・エントリーを管理する方法は 4 つあります。

- サーバー・エントリー名をアプリケーションにハードコーディングすることができます。ソース・コード内のサーバー・エントリー情報を変更できますが、エントリー名を有効にするには、そのアプリケーションを再コンパイルして再実行する必要があります。
- サーバー・エントリー名は、アプリケーションでアクセス可能な、サーバーの構成情報の **entryname** 属性として (**server** オブジェクトを使用して) 格納することができます。この方法は、再コンパイルする方法より便利ですが、さらに重要なのは、サーバーのエントリー名を、管理者も見ることができる、標準の (プラットフォームから独立した) 場所に置く点です。ただし、この方法を使用するには、アプリケーションの再起動が必要な場合があります。
- サーバー・エントリー名は、環境変数または引き数を介してアプリケーションに渡すことができます。この方法は再コンパイルする方法より効果的で、便利ですが、プラットフォームから独立していません。これは、オペレーティング・システムが異なると、別の方法が必要になる可能性があることを意味します。
- サーバー・エントリー名は、DCE 制御プログラムの **rpcentry** オブジェクトを使用して、 CDS 内で直接管理することができます。この手動の方法の場合は、アプリケーションの再コンパイルや再起動は必要ありません。

次の項では、CDS 内のサーバー・エントリーを手動で管理するための、 **rpcentry** オブジェクトの使用法について説明します。

CDS 内でのサーバー・エントリーの作成

サーバーの初期設定時、または初期設定後の構成時に、サーバーが CDS 内に自身のエントリーを作成することが多くあります。しかし、場合によっては、サーバー・エントリーを手動で作成したい場合があるかもしれません。作成された時点のサーバー・エントリーは空であり、インターフェース情報もバインディング情報も入っていません。

空のサーバー・エントリーを作成する理由の 1 つは、エントリーの所有権を確立するためです。サーバー・エントリーは、作成者によって所有されます。サーバーがエントリーを作成した場合、そのサーバーは後でそのエントリーを削除することもできます。管理者は、自分自身でエントリーを作成することによって、このような権利を獲得できます。その後、サーバーがそのバインディングを既存のサーバー・エントリーにエクスポートします (ACL によって許可されている場合)。

rpcentry create オペレーションを使用して、以下の例のように、空のサーバー・エントリーを作成します。この例では、 **/.:/subsys/applications/bbs_server** という名前のサーバー・エントリーを作成します。このオペレーションが成功するには、CDS ディレクトリー **/.:/subsys/applications** がすでに存在していなければなりません。

```
dcecp> rpcentry create /.:/subsys/applications/bbs_server
dcecp>
```

CDS からのサーバー・エントリーの削除

サーバー・エントリーは、一般に、安定したサーバー・バインディング情報を含んでいるので、削除されずにとどまる傾向があります。サーバーが短期間 (たとえば、一晩) 存在しなくなる場合でも、そのエントリーを削除することが実用的でないことがあります。しかし、サーバーが長期間存在しない場合は、サーバーのエントリーを CDS から削除することによって、存在しないサーバーをクライアントが使用しようと試みる無駄を回避することができます。

以下の例に示すように、**rpcentry delete** オペレーションを使用すれば、CDS からサーバー・エントリーを削除することができます。

```
dcecp> rpcentry delete
/.:/subsys/applications/bbs_server
dcecp>
```

CDS 内のサーバー・エントリーへのバインディング情報のエクスポート

サーバーの初期設定時または初期設定後の構成時に、サーバーは、通常、自身のバインディング情報を CDS にエクスポートします。しかし、バインディング情報が、何らかの理由あるいは事故で削除されていて、リストアしたい場合があります。あるいは、別のトランスポートが追加されていて、新しいトランスポートのバインディングをエクスポートしたい場合もあります。

rpcentry export オペレーションを使用して、サーバー・バインディング情報をサーバー・エントリーに手動でエクスポートすることができます。エントリーがまだ存在していない場合、**rpcentry export** オペレーションは、エントリーを作成します。ただし、それはディレクトリーがすでに存在し、しかも必要な許可を持っている場合に限られます。

次の例は、**/.:/subsys/applications/bbs_server** という名前のサーバー・エントリーへの、サーバーのバインディング情報のエクスポートを示しています。オブジェクト UUID は、 **bbs_server** によって使用されるデータ・ファイル・リソースを識別します。

```
dcecp> rpcentry export /.:/subsys/applications/bbs_server ¥
> -interface {458ffcbe-98c1-11cd-bd93-0000c08adf56 1.0} ¥
> -binding {ncacn_ip_tcp 130.105.1.227} ¥
> -object {76030c42-98d5-11cd-88bc-0000c08adf56}
dcecp>
```

CDS 内のサーバー・エントリーからのバインディング情報のインポート

アプリケーション・クライアント・プログラムは、CDS からサーバー・バインディング情報を自動的にインポートし、サーバーを検出しサーバーと通信するためにそれを使用することができます。しかし、管理者がバインディングをインポートしたい場合があります。たとえば、クライアントが CDS へのアクセスを持っていない場合でも、管理者が有効なバインディングを提供すれば、クライアントはサーバーと通信することができます。

以下のように、**rpcentry import** オペレーションを使用して、サーバーのバインディング情報を戻します。

```
dcecp> rpcentry import
/.:/subsys/applications/bbs_server ¥
> -interface {458ffcbe-98c1-11cd-bd93-0000c08adf56 1.0}
{ncacn_ip_tcp 130.105.1.227}
dcecp>
```

サーバー・エントリー内の情報の表示

クライアントがサーバーと通信できない場合、トラブルシューティングのステップとして、サーバー・エントリー内に含まれているバインディング情報を見たい場合があります。あるいは、サーバー・エントリーにオブジェクト UUID を追加した後、サーバー・エントリーを見落としていないかどうか調べたいことがあります。以下の例で示すように、**rpcentry show** オペレーションを使用して、サーバー・エントリー内の情報を表示することができます。戻される情報には、インターフェース識別子、サーバーに到達するための 2 つのバインディング、およびサーバーによって維持されるリソースのオブジェクト UUID が含まれます。

```
dcecp> rpcentry show /.:/subsys/applications/bbs_server
{458ffcbe-98c1-11cd-bd93-0000c08adf56 1.0
  {ncadg_ip_udp 130.105.1.227}
  {ncacn_ip_tcp 130.105.1.227}}
{76030c42-98d5-11cd-88bc-0000c08adf56}
dcecp>
```

CDS 内のサーバー・エントリーからのバインディング情報の削除

サーバー・エントリーからバインディング情報を削除しなければならない場合があります。サーバー・ホストがクラッシュした場合、そのサーバーは、CDS から自身のサーバー・エントリーを削除することができません。クライアントがこのような使用不能のサーバーとの通信を試みないように、対応するバインディングを手動で CDS からエクスポート解除する必要があります。 **endpoint delete** オペレーションの場合と異なり、このオペレーションは CDS からエントリー名を削除しません。

以下の例に示すように、**rpcentry unexport** オペレーションを使用して、サーバー・バインディング情報を削除します。オブジェクト UUID は、**unexport** オペレーションのオプションとしてそれを指定しない限り、サーバー・エントリーから削除されないことに注意してください。

```
dcecp> rpcentry unexport /.:/subsys/applications/bbs_server ¥
> -interface {458ffcbe-98c1-11cd-bd93-0000c08adf56 1.0}
dcecp>
dcecp> rpcentry show /.:/subsys/applications/bbs_server
{76030c42-98d5-11cd-88bc-0000c08adf56}
dcecp>
```

サーバーの作業負荷のバランスを取るためのグループ・エントリーの使用

クライアントがサーバー・バインディングについて CDS に照会する場合、この要求には、バインディングを探索するためのエントリー名が含まれます。クライアントが要求したサービスを 1 つのサーバーだけが提供する場合、CDS は、このサービスについてのすべてのクライアントの要求に対して、同じバインディングを戻します。限定されたクライアント要求の場合はこのモデルは適切に機能しますが、多くのクライアント要求が 1 つのサーバーに集中した場合は、サービス・ボトルネックを引き起こす可能性があります。アプリケーションは、多くのクライアント要求にサービスするための複数のサーバーを提供することによって、ボトルネックを回避することができます。サーバー・エントリー名だけの場合、各クライアントを特

定のサーバーに明示的に指示する必要があるので、複数のサーバー間でクライアント要求を均一に配分する便利な方法はありません。そのため、CDS では、複数のサーバーに渡ってクライアントの負荷を配分するための便利なメカニズムとして、グループ・エントリー を提供しています。

CDS グループ・エントリーは、関連するサーバーを、共通のグループ名のもとに集めます。グループ・エントリーには、一般に、サーバー・エントリーへのポインターである、メンバーが入っていますが、メンバーは他のグループ・エントリーを指すこともできます。クライアントがグループ・エントリーにバインディングを要求すると、CDS は、グループ・エントリーに入っているポインターの 1 つをランダムに戻します。ランダムに取り上げられたエントリーが別のグループ・エントリーである場合、CDS はそれを戻しません。その代わり、CDS はそのグループで別のランダム・メンバーを取り上げ、サーバー・エントリーが戻されるまで、それを続けます。このモデルの場合、どのグループ・メンバーもクライアントの要求に対してサービスできなければなりません。 図3 は、グループ・エントリーに、他のグループを指すメンバー、およびサーバー・エントリーを指すメンバーがどのように入っているかを示したものです。

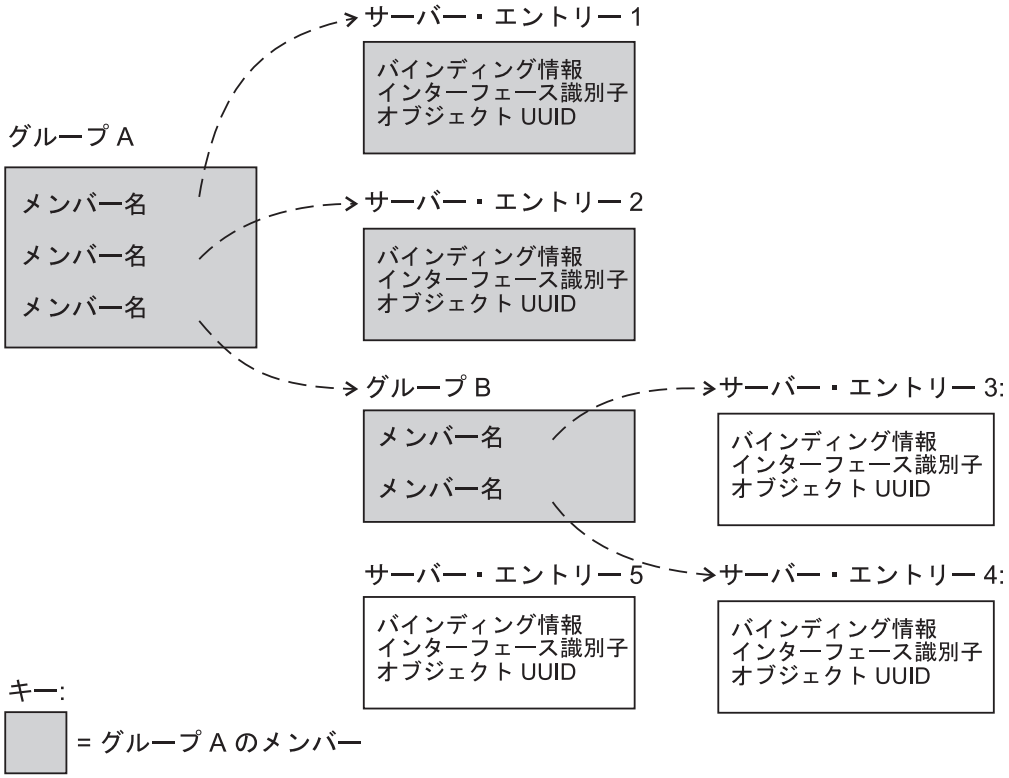


図3. グループの可能なマッピング

ここで、グループ・エントリーが作業負荷のバランスを取る方法を見てみます。 12 台の同一のレーザー・プリンターが、3 つの部門に平均して分散している組織について考えます。次のグループ・エントリー例は、各グループ・エントリー名が、各部門に割り当てられている 4 つのプリンターのいずれか 1 つを戻す方法を示しています。

```
Group entry name: /.:/admin/finance/accts_payable_printers
  /.:/admin/finance/accts_payable/laser_10
  /.:/admin/finance/accts_payable/laser_11
  /.:/admin/finance/accts_payable/laser_12
  /.:/admin/finance/accts_payable/laser_13
Group entry name: /.:/admin/finance/accts_receivable_printers
  /.:/admin/finance/accts_receivable/laser_10
  /.:/admin/finance/accts_receivable/laser_11
  /.:/admin/finance/accts_receivable/laser_12
  /.:/admin/finance/accts_receivable/laser_13
Group entry name: /.:/admin/finance/payroll_printers
 /.:/admin/finance/payroll/laser_10
 /.:/admin/finance/payroll/laser_11
 /.:/admin/finance/payroll/laser_12
 /.:/admin/finance/payroll/laser_13
```

次のグループ・エントリー例に示すように、そのグループ名を他のグループのグループ・エントリーに追加することによって、一時的に、ある部門のプリンターを別のグループが使用できるようにすることができます。

```
Group entry name: /.:/admin/finance/accts_payable_printers
  /.:/admin/finance/accts_payable/laser_10
  /.:/admin/finance/accts_payable/laser_11
  /.:/admin/finance/accts_payable/laser_12
  /.:/admin/finance/accts_payable/laser_13
  /.:/admin/finance/accts_receivable_printers
```

上記の例の構成は、買掛管理のクライアントが、売掛管理のプリンターの稼働時間のうち 20% を使用できることを意味します。より高いパーセンテージで使用できるようにするには、グループ名ではなく、サーバー・エントリー名を追加します。次のグループ・エントリー例は、買掛管理のクライアントが売掛管理のプリンターの稼働時間のうち 50% を使用できる状態を示しています。グループ名を何度も組み込むことによって使用率を上げようとすると、エラーになるため、これを行わないでください。

```
Group entry name: /.:/admin/finance/accts_payable_printers
  /.:/admin/finance/accts_payable/laser_10
  /.:/admin/finance/accts_payable/laser_11
  /.:/admin/finance/accts_payable/laser_12
  /.:/admin/finance/accts_payable/laser_13
  /.:/admin/finance/accts_receivable/laser_10
  /.:/admin/finance/accts_receivable/laser_11
  /.:/admin/finance/accts_receivable/laser_12
  /.:/admin/finance/accts_receivable/laser_13
```

アプリケーション・サーバーが CDS 内の自身のグループ・エントリーを管理できますが、グループ・エントリー内のサーバー情報を手動で追加、削除、または変更する方がより便利 (かつ単純) な場合があります。サーバー・エントリーの管理と同様に、CDS 内のグループ・エントリーを管理する方法がいくつかあります。

- グループ・エントリー名を、アプリケーション内にハード・コーディングすることができます。ソース・コード内のグループ・エントリー情報を変更できますが、エントリー名を有効にするには、アプリケーションを再コンパイルして再実行する必要があります。
- 環境変数または引き数を介して、グループ・エントリー名をアプリケーションに渡すことができます。これらは再コンパイルよりも便利な方法ですが、いずれの方法を使用するにも、アプリケーションの再起動が必要になる可能性があります。

- グループ・エントリー名は、DCE 制御プログラムの **rpcgroup** オブジェクトを使用して、 CDS 内で直接管理することができます。この手動の方法の場合は、アプリケーションの再コンパイルや再起動は必要ありません。

次の項では、CDS 内のグループ・エントリーを手動で管理するための、 **rpcgroup** オブジェクトの使用法について説明します。

CDS 内での新しいグループ・エントリーの作成

rpcgroup create オペレーションを使用して、 CDS 内に空のグループ・エントリーを作成することができます。グループの作成は、グループ・エントリーを最初に使用するアプリケーションによって行われることが多いのですが、エントリーを自分で作成すれば、そのエントリーの所有者となります。エントリーの所有者は、エントリー内の情報をエクスポートし管理できる人を完全に制御することができます。

CDS 内に空のグループ・エントリーを作成するには、以下の例のように、**rpcgroup create** オペレーションを使用します。

```
dcecp> rpcgroup create
/.:/subsys/applications/admin_bbs_servers
dcecp>
```

CDS 内でのグループ・エントリーへのメンバーの追加

rpcgroup add オペレーションを使用して、メンバーをグループ・エントリーに追加することができます。グループ・エントリーが存在しない場合は、このオペレーションは、グループ・エントリーを作成してから、メンバーを追加します。メンバーはサーバー・エントリーでも、他のグループ・エントリーでもかまいません。追加したメンバーが実際に存在するかどうかを検査するオペレーションはないことに注意してください。このため、サーバーを立ち上げて実行する前でも、ネームスペースを構成することができます。

CDS 内の **/.:/subsys/applications/admin_bbs_servers** グループ・エントリーにメンバーを追加するには、以下の例のように、**rpcgroup add** オペレーションを使用します。

```
dcecp> rpcgroup add /.:/subsys/applications/admin_bbs_servers ¥
> -member /.:/subsys/applications/bbs_server4
dcecp>
```

グループ・エントリーのメンバーの表示

rpcgroup list オペレーションを使用して、グループ・エントリーのメンバーをリストすることができます。これは、トラブルシューティングのため、またはグループ・エントリー内でサーバーがどのように分散しているかを表示する場合に役立ちます。

CDS 内のグループ・エントリーのメンバーをリストするには、以下の例に示すように、**rpcgroup list** オペレーションを使用します。この例では、グループ **/.:/subsys/applications/admin_bbs_servers** のメンバーをリストします。

```
dcecp> rpcgroup list /.:/subsys/applications/admin_bbs_servers
/.../my_cell.goodco.com/subsys/applications/bbs_server3
/.../my_cell.goodco.com/subsys/applications/bbs_server4
dcecp>
```

CDS 内のグループ・エントリーからのバインディング情報のインポート

アプリケーション・クライアント・プログラムは、 CDS からサーバー・バインディング情報を自動的にインポートし、サーバーを検出しサーバーと通信するためにそれを使用することができます。しかし、管理者がバインディングをインポートしたい場合があります。クライアントが CDS へのアクセスを持たない場合でも、管理者が有効なバインディングをクライアントに提供すれば、クライアントはサーバーと通信することができます。

rpcgroup import オペレーションを使用して、サーバーのバインディング情報を戻すことができます。以下の例に示すように、**-interface** オプションを使用して、インターフェースを指定しなければなりません。

```
dcecp> rpcgroup
import /.:/subsys/applications/admin_bbs_servers ¥
> -interface {458ffcbe-98c1-11cd-88bc-0000c08adf56 1.0}
{ncacn_ip_tcp 130.105.1.227}
dcecp>
```

バインディング情報を詳細に指定するために、 **-version** および **-object** などのオプションを使用することができます。戻されるバインディングの個数を制限するには、 **-max** オプションを使用します。

CDS 内のグループ・エントリーからのメンバーの削除

時が経過するうちに、組織変更によって、DCE セル内のサーバーの再配置が必要になることがあります。たとえば、サーバー・エントリーをあるグループ・エントリーから別のグループ・エントリーに移動したくなるかもしれません。

rpcgroup remove オペレーションを使用して、グループから 1 つまたは複数のメンバーを削除します。以下の例は、グループ **/.:/subsys/applications/admin_bbs_servers** から **bbs_server3** を削除します。

```
dcecp> rpcgroup remove /.:/subsys/applications/admin_bbs_servers ¥
> -member /.../my_cell.goodco.com/subsys/applications/bbs_server3
dcecp> rpcgroup list /.:/subsys/applications/admin_bbs_servers
/.../my_cell.goodco.com/subsys/applications/bbs_server4
/.../my_cell.goodco.com/subsys/applications/bbs_server5
/.../my_cell.goodco.com/subsys/applications/bbs_server6
dcecp>
```

CDS からのグループ・エントリーの削除

組織変更またはサーバーの再配置によって、いくつかのグループが古くなることがあります。 CDS からグループ・エントリーを削除したい場合は、 **rpcgroup delete** オペレーションを使用します。以下の例は、**/.:/subsys/admin/temporaries/wp_services** という古いグループ・エントリーを CDS から削除することを示しています。

```
dcecp> rpcgroup delete /.:/subsys/admin/temporaries/wp_services
dcecp>
```

サーバーでの直接クライアント検索のためのプロファイルの使用

グループ・エントリーは、複数の使用可能なサービスからのランダム選択をクライアントに対して提供します。グループ・エントリーは、負荷バランスとリソース割り振りには役立ちますが、そのランダムな特性は、微調整する場合の妨げになりま

す。さらに、グループ・エントリーでは、特定のクライアントが使用するための、サーバーの優先順位づけの手段を提供していません。

プロファイルはサーバーを編成するための補完的な手段を提供しますが、これは、プロファイル・メンバーの探索順序には優先順位づけができるためです。 (以前の DCE バージョンでは、これらのメンバーをエレメント と呼んでいました。) メンバーは、以下の情報を提供することによってサーバーを識別します。

- インターフェース識別子

  このフィールドは、プロファイルのキーになります。このインターフェース識別子は、インターフェース UUID とインターフェース・バージョン番号からなります。

- メンバー名

  以下のいずれかの種類のディレクトリー・サービス・エントリーのエントリー名。

  1. 要求された RPC インターフェースを提供するサーバーのサーバー・エントリー
  2. 要求された RPC インターフェースに対応するグループ
  3. プロファイル

- 優先順位の値

  優先順位の値 (0 が最高の優先順位で、7 が最低の優先順位) は、同じ優先順位のメンバーをランダムに選択する探索順序を決定するために、プロファイル・メンバーの作成者が指定します。

- 注釈ストリング

  注釈ストリングによって、プロファイル・メンバーの目的を識別することができます。注釈は任意のテキスト情報です。たとえば、インターフェース識別子に関連するインターフェース名、あるいはグループに関連するサービスまたはリソースの説明などです。

  注釈ストリングは、インターフェース識別子フィールドと異なり、探索キーではありません。

プロファイルには柔軟性があり、それにはサーバー・エントリー、グループ、および他のプロファイルを指すことができるメンバーが入っています。プロファイルには、*デフォルト・プロファイル・メンバー* と呼ばれる、特殊なメンバーを入れることもできます。このオプションのメンバーは、デフォルト・プロファイルを指す必要があり、通常、このプロファイルは広範囲のバックアップ・プロファイルであり、オーガニゼーション内のほとんどのユーザーの要求を満たすものです。 146ページの図4 は、プロファイルの可能なマッピングを示しています。

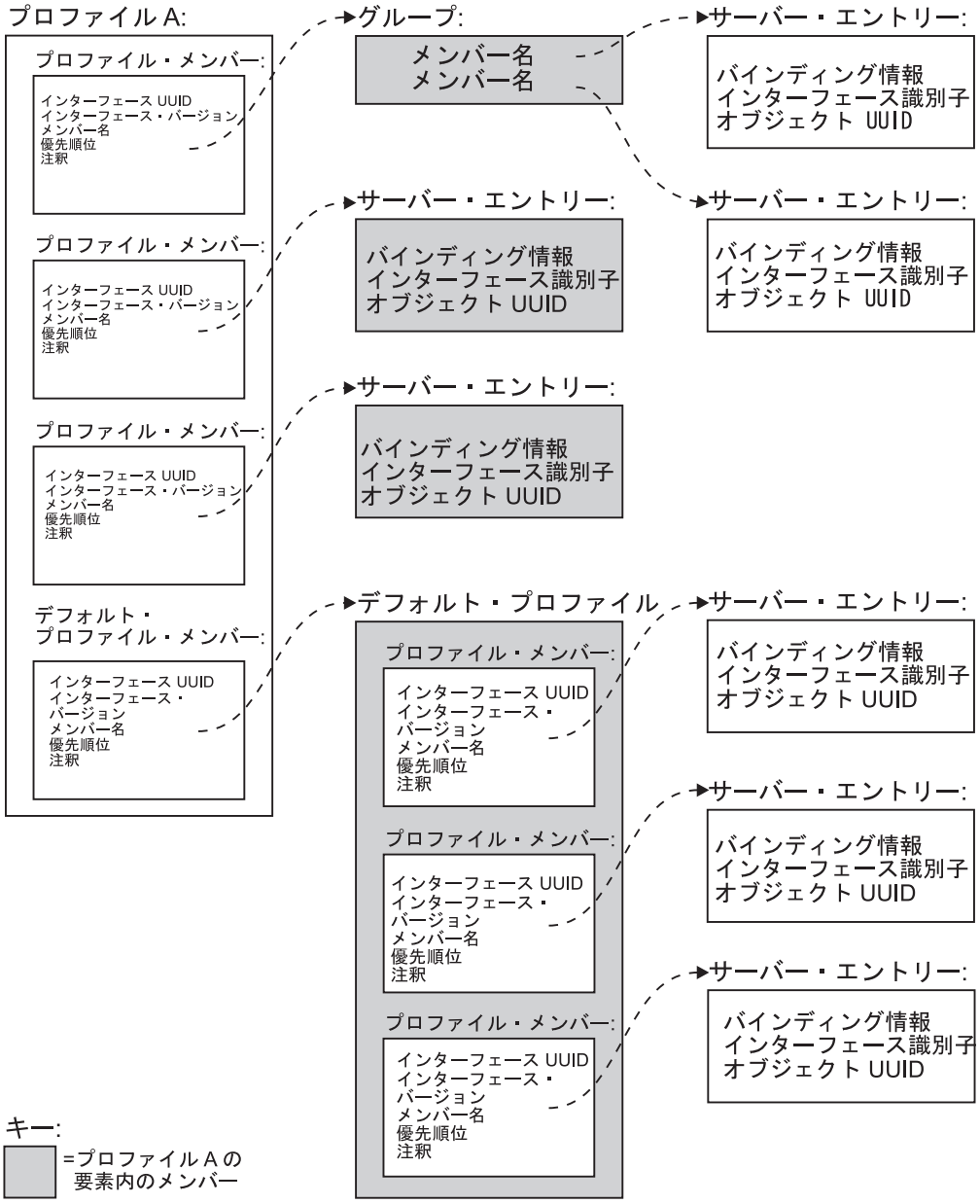


図 4. プロファイルの可能なマッピング

プロファイルの機能を理解するために、グループ・エントリーについての上記の説明からプリンターの例を取り上げます。次のプロファイル・エントリーの例は、プロファイルを使用してクライアントへの近接度に基づいてリソースの優先順位を付ける、1 つの方法を示しています。

図では、3 人のユーザーは個別のプリンター・プロファイルを持っています。これらのプロファイルは、最も近いプリンターのサーバー・エントリーを最初に戻します。たとえば、ユーザー **John** は **laser_20** に最も近いので、プロファイル優先順位 **1** がそのバインディングを最初に戻します。 **John** は **laser_23** から最も遠いので、プロファイル優先順位 **4** がそのバインディングを最後に戻します。

```
Profile entry name: /.:/admin/finance/accts_receivable_printers/johns_profile
  /.:/admin/finance/accts_receivable/laser_20  1
  /.:/admin/finance/accts_receivable/laser_21  2
```

```
  /.:/admin/finance/accts_receivable/laser_22  3
  /.:/admin/finance/accts_receivable/laser_23  4
Profile entry name: /.:/admin/finance/accts_receivable_printers/pats_profile
  /.:/admin/finance/accts_receivable/laser_20  3
  /.:/admin/finance/accts_receivable/laser_21  4
  /.:/admin/finance/accts_receivable/laser_22  2
  /.:/admin/finance/accts_receivable/laser_23  1
Profile entry name: /.:/admin/finance/accts_receivable_printers/wills_profile
  /.:/admin/finance/accts_receivable/laser_20  2
  /.:/admin/finance/accts_receivable/laser_21  1
  /.:/admin/finance/accts_receivable/laser_22  3
  /.:/admin/finance/accts_receivable/laser_23  4
```

この例を終える前に、自分の部門のサーバーが別の部門によって過剰に使用されている場合を考えてみましょう。自分のサーバーを指している他の部門のプロファイル内の当該サーバーの優先順位の値を下げることによって、他の部門による使用を制限することができます。

アプリケーション・サーバーが CDS 内の自身のプロファイル・エントリーを管理できるのと同じように、自身のプロファイル・エントリーを管理することもできます。しかし、プロファイル・エントリー内のサーバー情報を手動で追加、削除、または変更する方がより便利 (かつ単純) な場合があります。サーバー・エントリーとグループ・エントリーの管理と同様に、 CDS 内のプロファイル・エントリーを管理する方法がいくつかあります。

- プロファイル・エントリー名をアプリケーション内にハード・コーディングすることができます。ソース・コード内のプロファイル・エントリー情報を変更できますが、エントリー名を有効にするには、そのアプリケーションを再コンパイルして再実行する必要があります。
- 環境変数または引き数を介して、プロファイル・エントリー名をアプリケーションに渡すことができます。これらの方法は再コンパイルより便利ですが、いずれの方法を使用するにも、アプリケーションの再起動が必要になる可能性があります。
- プロファイル・エントリー名は、 DCE 制御プログラムの **rpcprofile** オブジェクトを使用して、CDS 内で直接、管理することができます。この手動の方法の場合は、アプリケーションの再コンパイルや再起動は必要ありません。

以下の項では、CDS 内のプロファイル・エントリーを手動で管理するための、**rpcprofile** オブジェクトの使用法について説明します。

新しいプロファイルの作成

rpcprofile create オペレーションを使用して、 CDS 内に空のプロファイル・エントリーを作成することができます。プロファイルの作成は、プロファイル・エントリーを最初に使用するアプリケーションによって行われることが多いのですが、エントリーを自分で作成すれば、エントリーの所有者となります。エントリーの所有者は、エントリー内の情報をエクスポートし管理できる人を完全に制御することができます。

CDS 内に空のプロファイル・エントリーを作成するには、以下の例のように、**rpcprofile create** オペレーションを使用します。

```
dcecp> rpcprofile create
/.:/subsys/applications/admin_group_profile
dcecp>
```

プロファイル・メンバーの追加

rpcprofile add オペレーションを使用して、メンバーをプロファイル・エントリーに追加することができます。プロファイル・エントリーが存在しない場合は、このオペレーションはプロファイル・エントリーを作成してから、メンバーを追加します。メンバーはサーバー・エントリーでも、他のプロファイル・エントリーでもかまいません。

CDS 内の **/.:/subsys/applications/wards_profile** プロファイル・エントリーにメンバーを追加するには、以下の例のように、**rpcprofile add** オペレーションを使用します。この例では、優先順位 **2** のサーバー・エントリー **/.:/subsys/applications/bbs_server3** を追加しています。

```
dcecp> rpcprofile add /.:/subsys/applications/wards_profile ¥
> -member /.:/subsys/applications/bbs_server3 ¥
> -interface {458ffcbe-98c1-11cd-88bc-0000c08adf56 1.0} ¥
> -priority 2
dcecp>
```

プロファイル・エントリーのメンバーの表示

rpcprofile list オペレーションを使用して、プロファイル・エントリーのメンバーを単純にリストすることができます。これは、トラブルシューティングの場合、またはプロファイル・エントリー内でサーバーがどのように分散しているかを表示する場合に便利です。

CDS 内のプロファイル・エントリーのメンバーをリストするには、以下の例に示すように、**rpcprofile list** オペレーションを使用します。この例では、プロファイル **/.:/subsys/applications/admin_group_profile** のメンバーをリストします。

```
dcecp> rpcprofile list /.:/subsys/applications/wards_profile
/.../my_cell.goodco.com/subsys/applications/admin_bbs_servers
/.../my_cell.goodco.com/subsys/applications/bbs_server
dcecp>
```

rpcprofile show オペレーションを使用して、プロファイル・エントリーに保管されている完全な情報を表示することができます。これは、そのメンバーに関連する優先順位とインターフェース UUID を表示します。以下の例は、 **/.:/cell-profile** という名前のプロファイルの中に含まれるすべての情報を表示します。

```
dcecp> rpcprofile show /.:/cell-profile
{{d46113d0-a848-11cb-b863-08001e046aa5 2.0} /.../cell.co.com/sec 0 rs_bind}
{{0d7c1e50-113a-11ca-b71f-08001e01dc6c 1.0} /.../cell.co.com/sec-v1 0 secidmap}
{{8f73de50-768c-11ca-bffc-08001e039431 1.0} /.../cell.co.com/sec 0 krb5rpc}
{{b1e338f8-9533-11c9-a34a-08001e019c1e 1.0} /.../cell.co.com/sec 0 rpriv}
{{b1e338f8-9533-11c9-a34a-08001e019c1e 1.1} /.../cell.co.com/sec 0 rpriv}
{{6f264242-b9f8-11c9-ad31-08002b0dc035 1.0} /.../cell.co.com/lan-profile 0 LAN}
{{4d37f2dd-ed43-0000-02c0-37cf2e000001 4.0} /.../cell.co.com/fs 0 fs}
dcecp>
```

CDS 内のプロファイル・エントリーからのバインディング情報のインポート

アプリケーション・クライアント・プログラムは、 CDS からサーバー・バインディング情報を自動的にインポートし、サーバーを検出しサーバーと通信するためにそれを使用することができます。しかし、管理者がバインディングをインポートしたい場合があります。クライアントが CDS へのアクセスを持たない場合でも、管理者が有効なバインディングをクライアントに提供すれば、クライアントはサーバーと通信することができます。

rpcprofile import オペレーションを使用して、サーバーのバインディング情報を戻すことができます。以下の例に示すように、**-interface** オプションを使用して、インターフェースを指定しなければなりません。

```
dcecp> rpcprofile import /.:/subsys/applications/wards_profile ¥
> -interface {458ffcbe-98c1-11cd-88bc-0000c08adf56 1.0}
{ncacn_ip_tcp 130.105.1.202}
{ncacn_ip_tcp 130.105.1.227}
dcecp>
```

バインディング情報を詳細に指定するために、 **-version** および **-object** などのオプションを使用することができます。戻されるバインディングの数を制限するには、以下の例に示すように、**-max** オプションを使用します。

```
dcecp> rpcprofile import /.:/subsys/applications/wards_profile ¥
> -interface {458ffcbe-98c1-11cd-88bc-0000c08adf56 1.0} ¥
> -max 1
{ncacn_ip_tcp 130.105.1.202}
dcecp>
```

CDS 内のプロファイル・エントリーからのメンバーの削除

時が経過するうちに、組織変更によって、DCE セル内のサーバーの再配置が必要になることがあります。たとえば、サーバー・エントリーをあるプロファイル・エントリーから別のプロファイル・エントリーに移動する必要があるかもしれません。

rpcprofile remove オペレーションを使用して、プロファイルから 1 つまたは複数のメンバーを削除します。以下の例では、**rpcprofile remove** オペレーションが、メンバー **/.:/subsys/applications/admin_bbs_servers** をプロファイル **/.:/subsys/applications/wards_profile** から削除します。

```
dcecp> rpcprofile remove /.:/subsys/applications/wards_profile ¥
> -member /.:/subsys/applications/admin_bbs_servers ¥
> -interface {458ffcbe-98c1-11cd-88bc-0000c08adf56 1.0}
dcecp>
```

CDS からのプロファイル・エントリーの削除

組織変更またはサーバーの再配置によって、いくつかのプロファイルが古くなることがあります。 CDS からプロファイル・エントリーを削除したい場合は、**rpcprofile delete** オペレーションを使用します。以下の例は、**/.:/subsys/admin/temporaries/74232_profile** という古いプロファイル・エントリーの CDS からの削除を示しています。

```
dcecp> rpcprofile delete /.:/subsys/admin/temporaries/74232_profile
dcecp>
```

クライアントの管理

この章では、これまで、サーバーの管理問題に焦点を当ててきました。いくつかのサーバー・オペレーションを制御する方法、およびクライアントが検出できる CDS 内およびホスト・エンドポイント・マップ内にサーバー・バインディング情報を格納する方法を見てきました。この項では、アプリケーション・クライアントの管理要件について説明します。クライアント管理は非常に単純ですが (関連するオペレーションは 2 つだけです)、クライアントとサーバーを一緒に作業させるための必須ステップです。

CDS が、サーバー・エントリーの形でサーバー・バインディング情報を格納する、ディレクトリーの階層システムであることは説明しました。また、CDS が、クライアントを適切なサーバーに仕向ける方法として、グループ・エントリーとプロファイル・エントリーを提供することも説明しました。しかし、クライアントは、サーバーの探索をどこから開始するかをどのようにして知るのでしょうか。

この章の前半で説明したように、サーバーは、インターフェースとそのバインディングを CDS 内に登録します。各インターフェースとバインディングの組み合わせは、サーバー・エントリー名のもとに登録されます。クライアントがリモート・プロシージャー・コールを行う場合、サーバー・エントリー名 (あるいはグループ・エントリー名またはプロファイル・エントリー名) を、リモート・プロシージャーを提供するインターフェースの UUID とともに、CDS に渡します。 CDS は、このサーバー・エントリー名 (あるいはグループ・エントリー名またはプロファイル・エントリー名) を、クライアントから渡されたものと一致するインターフェース UUID とバージョンが含まれているバインディングを探索するための開始点として、使用します。この方法は、クライアントが、サーバーが使用するサーバー・エントリー名 (あるいはグループ名またはプロファイル名) をすでに獲得しているものと推定しています。

クライアントに適切なサーバー・エントリー名を使用させるには、次の 2 つのステップからなるプロセスが必要です。

1. クライアントが使用するエントリー名を判別する。
2. 名前をクライアント・プログラムに渡す。

クライアントは提供された名前をそのまま使用することに注意してください。クライアント・プログラムは、名前がサーバー・エントリー名またはグループ・エントリー名またはプロファイル・エントリー名であるかどうかを区別することができません。クライアントには、これらの名前はすべて同じように見え、同じように動作します。

エントリー名の判別

クライアント・プログラムを構成する場合にサーバーがエクスポートするエントリー名を提供するためには、それを知っていなければなりません。ここでは、この名前をエントリー名とだけ呼んでいますが、サーバー・エントリー名またはグループ・エントリー名またはプロファイル・エントリー名である可能性があります。アプリケーションの資料を参照して、使用すべきエントリーの種類を決定してください。

アプリケーションのサーバー部分とクライアント部分をインストールし構成している場合は、サーバーを構成する時に、サーバーのエントリー名を控えておいてください。

サーバーをインストールまたは構成しない場合 (たとえば、サーバーがすでにインストール済みの場合) は、使用する名前を判別するための検出作業が必要になるかもしれません。検出場所はいくつかあります。

この章の前半で説明したサーバー制御機能をサーバーが使用している場合は、エントリー名を表示するために、おそらく **server show** オペレーションを使用することができます。もちろん、これは、サーバーが常駐しているホスト上のサーバーの

オブジェクト名を知っていなければならないことを意味します。 **server catalog** オペレーションを使用して、ホスト上のすべてのサーバー・オブジェクト名を表示することができます。以下の例は、ホスト **silver** 上に構成されたすべてのサーバー・オブジェクトをリストします。 **server show** オペレーションは、**info_server** プログラムで使用されたエントリー名を表示します。

```
dcecp> server catalog /.:/hosts/silver
/.../my_cell.goodco.com/hosts/silver/config/srvrconf/video_clip
/.../my_cell.goodco.com/hosts/silver/config/srvrconf/info_server
dcecp> server show /.:/hosts/silver/config/srvrconf/info_server
{uuid 6d5e7184-71b7-11cd-a205-08000925634b}
{program {/usr/local/bin/infosrv}}
{arguments {-brief}}
{prerequisites {}}
{keytabs {}}
{entryname {/.:/subsys/applications/info_server_1}}
{services {}}
{principals {}}
{starton {explicit failure}}
{uid 1423}
{gid 1000}
{dir {/tmp}}
dcecp>
```

サーバーがブート・プログラムまたはある種のスクリプトから開始された場合には、そのプログラムまたはスクリプトの中で名前 (サーバーは、複数のインターフェースをエクスポートした場合、複数の名前を使用することがあります) を探してください。名前は、以下の例に示すように、サーバーを開始するコマンドへの引き数として与えます。

```
infosrv /.:/finance/operations/infoserv
```

サーバー側がそのエントリー名を容易に表示できない場合は、他のクライアント・プログラムが使用しているエントリーを調べてみてください。クライアント・プログラムは、ブート・プログラムまたはある種のスクリプトから開始されることが多く、一般に、エントリー名がクライアントを開始するためのコマンドへの引き数として与えられます。これらのコマンドは、上記のサーバー起動コマンドの例に示した同じモデルに従う場合がしばしばあります。

クライアントへのエントリー名の提供

きわめて単純なクライアントは、その中にサーバー・エントリー名をコーディングできるので、エントリー名を渡す必要がない場合があります。しかし、多くの場合、クライアント・プログラムの開始時に、エントリー名を与える必要があります。この方法は、必要な場合に、別のエントリー名を簡単に使用することができるので、エントリー名をハード・コーディングする方法よりも柔軟性があります。

クライアント構成資料には、クライアントに名前を渡す方法に関する指示が含まれていなければなりません。 1 つの方法では、クライアントを開始するためのコマンドと、適切なサーバー・エントリー名を含んだ引き数とが一緒に入っているスクリプトまたはバッチ・ファイルを使用します。次の例は、クライアントを開始するシェル・スクリプト内のコマンド引き数として渡される、サーバー・エントリー名を示しています。

```
# Shell Script to start the InfoClient application
infoclient /.:/finance/operations/InfoServ_profile
```

あるいは、サーバー・エントリー名を環境変数 (UNIX システムでは **RPC_DEFAULT_ENTRY** と呼ばれる) に格納することができます。次の例は、この変数を定義してからクライアントを呼び出す、シェル・スクリプトを示しています。

```
#! /bin/sh
# Shell Script to start the InfoClient application
export RPC_DEFAULT_ENTRY=/.:/finance/operations/InfoServ_profile
infoclient
```

第4部 セル・ディレクトリー・サービス

第11章 DCE ディレクトリー・サービスの紹介

分散処理では、従来のコンピューティング環境の中では 1 つのシステムで行われていた作業を分割して処理する、複数のシステム間の対話が必要になります。このようなネットワーク全体で作業を行う環境の結果生じる 1 つの課題は、ネットワーク内のあらゆる場所の人々やリソースを識別し探し出すために、全体として一貫性のある方法が必要になることです。

DCE ディレクトリー・サービスを使用すると、ネットワーク内の人々とコンタクトしたり、またディスク、印刷キュー、およびサーバーなどのリソースを、その物理的位置を知らなくても、使用できるようになります。ディレクトリー・サービスは、ある人の名前を知らせれば電話番号を教えてくれる、電話番号案内サービスとよく似ています。人間、サーバー、またはリソースの名前を指定すれば、このサービスはネットワーク・アドレスやその名前に関連するその他の情報を戻すことができます。

DCE ディレクトリー・サービスは、アドレスおよびその他の関連情報を、その名前の*属性* として保管します。たとえば、属性には、組織の構成単位の名前 (European Sales など)、場所 (ビルディング A の 1 階など)、あるいは電話番号などを含めることができます。

DCE コンポーネントが DCE ディレクトリー・サービスを使用する方法

DCE ディレクトリー・サービスは、アプリケーションが信頼してその長所を利用できる基本的サービスです。この節では、他の DCE コンポーネントが DCE ディレクトリー・サービスをどのように使用するかを説明します。

DCE リモート・プロシージャー・コール (RPC) インターフェースを使用すると、クライアント / サーバー・モデルにしたがった分散アプリケーションの開発と使用が容易になります。 RPC モデルでは、*クライアント* はリモート・プロシージャー・コールを行うプログラムであり、*サーバー* はそのプロシージャーを実行するプログラムです。 DCE RPC ソフトウェアは、 RPC サーバーのアドレスおよびサポートするインターフェースに関する情報をディレクトリー・サービスに格納します。

ある特定のサーバーを呼び出したい場合、RPC クライアントは、ディレクトリー・サービスにそのサーバーとの連絡に必要な情報を照会することができます。ディレクトリー・サービス内の名前を付けられたある特定のリソースにアクセスしたい場合、クライアントはその特定の名前を照会することができます。クライアント・アプリケーションが、必要なサービスのタイプ (たとえば、C コンパイラー、プリンター、あるいは従業員情報) は知っているが、特定のサーバーのアドレスを知らない場合、ディレクトリー・サービスを使用してその情報を見つけることもできます。

DCE セキュリティー・サービス (ユーザーがログインするときにその ID を検査する) は、ディレクトリー・サービスを使用してその認証サーバーのアドレスを格納します。

分散ファイル・サービス (DFS) は、ファイル・セット (ファイルの論理グループ) に対して位置サービスを提供するため、ユーザーはリモート・ファイルがローカル・システムにある場合と同じように、それにアクセスすることができます。 DFS は、DCE ディレクトリー・サービスを使用して、そのファイル・セット位置サーバーに連絡する方法を見つけます。

分散タイム・サービス (DTS) は、ネットワーク内のシステム・クロックの同期化を行います。クロックの同期化は、複数のシステムにわたってイベントが発生する順序を追跡する必要のある分散アプリケーションにおいては重要です。 DTS は、DCE ディレクトリー・サービスを使用して、そのタイム・サーバーを突き止める方法を見つけます。

DCE ディレクトリー・サービスを使用する方法

DCE 管理者を除き、ディレクトリー・サービスを使用する人は、通常、アプリケーション・インターフェースを介して、間接的にディレクトリー・サービスを使用します。アプリケーションは、リソースの名前を作成するユーザーに代わって、ディレクトリー・サービスと対話し、その後はその名前でこのリソースを参照できます。次に示す例は、事実も仮定も含めて、ユーザーがディレクトリー・サービスを使用できるいくつかの方法について説明しています。

- ユーザーが新しい文書でスペル・チェック・アプリケーションを起動します。このアプリケーションには、ユーザーのローカル・システム上にある DCE RPC クライアント・コードが含まれています。 RPC クライアントは、ディレクトリー・サービスと連絡をとり、使用可能なスペル・チェック・サーバーに関する情報を要求します。ディレクトリー・サービスは、サーバーのアドレス、サーバーが通信に使用するプロトコル・タイプ、およびインターフェースを表すユニバーサル・ユニーク識別子 (UUID) を戻します。この情報を使用して RPC クライアントがサーバーに対してリモート・コールを行うと、サーバーはユーザーの文書のスペル・チェックを行います。ユーザーは、スペル・チェック・プログラムを使用しても、それがディレクトリー・サービスを呼び出して、リモート・サーバーと対話していることには気付きません。
- システムにログインするユーザーは、名前とパスワードを入力します。ディレクトリー・サービスは、ログイン・プログラムが認証データベースの中のユーザー ID を検査する認証サーバーを探し出すのを援助します。
- ユーザーはファイル指定を入力します。ディレクトリー・サービスは、DFS ファイル・セット位置データベースのアドレスを提供します。これには、ユーザーがファイルにアクセスできるようにするサーバーのネットワーク・アドレスが入っています。
- ユーザーがコンピューター会議または電子掲示板の名前を入力すると、ディレクトリー・サービスは、アプリケーションを会議サービスに接続できるようにするためのアドレスを提供します。
- プリンター機能の名前またはプリンターに関する何らかの情報を入力すると、ユーザーはプリンターのネットワーク・アドレスを知ることができます。たとえば、ユーザーは、現在使用可能な、最も近くにある最も速いカラー・プリンターのアドレスを知りたい場合があります。
- ユーザーには、販売部門の従業員からの情報が必要です。ユーザーは、その従業員のラストネームが Wong であることは思い出しましたが、下の名前が思い出せ

ません。従業員探索アプリケーションにラストネームと部門名を入力すれば、ユーザーは、販売部門のすべての Wong さんに関する情報を見つけるためにディレクトリー・サービスを検査して、その従業員に連絡する方法を見つけ出すことができます。

- ユーザーは問題追跡データベースに報告書を入力します。データベースは最近、新しいノードに移動されましたが、ユーザーは常にそのデータベースを名前だけで参照していたので、変更には気付きません。ディレクトリー・サービスは、現行のネットワーク・アドレスを格納し、それを問題追跡アプリケーションおよびそれを必要とするその他のすべてのアプリケーションに提供します。

この章の残りの部分では、 DCE ディレクトリー・サービス環境がセルに関連してどのように機能するかについて説明します。ここではディレクトリー・サービスの主要コンポーネントを紹介します。これには、ローカルとグローバルのネーミング環境間のゲートウェイである、セル・ディレクトリー・サービス (CDS) およびグローバル・ディレクトリー・エージェント (GDA) があります。また、この章では、DCE テクノロジー・オファリングには含まれていないグローバル名サービスである、ドメイン・ネーム・システム (DNS) および LDAP サーバーに対する DCE サポートについても説明します。

ディレクトリー・サービスとセル環境

この項では、以下のような DCE ネーミング環境の主要コンポーネントを紹介し、こうしたコンポーネントとセルとの関係について説明します。

- CDS
- DNS
- GDS
- LDAP クライアント・サーバー
- GDA

CDS は、セル内部 (セル内) におけるリソースのネーミングや使用のための、整合性があり、場所に依存しない方法を提供する、高性能分散サービスです。また、CDS はセル相互間 (セル間) の通信にも使用することができます。

DNS は、DCE がサポートし、現在広く使用されているグローバル・ネーム・サービスです。多くのネットワークはインターネット・ホスト名のネーム・サービスとして現在、主に DNS を使用しています。 DNS は DCE テクノロジー・オファリングには含まれていませんが、ディレクトリー・サービスには、セルが DNS を介して相互運用するためのサポートが含まれています。 GDS X.500 サーバーは、X.500 クライアントからディレクトリー・アクセス・プロトコル (DAP) を受け取り、そのディレクトリーのオブジェクトにアクセスします。 DCE では、サーバーは GDS サーバー、クライアントは GDS クライアントです。 GDA は、XDS/XOM API を介して GDS クライアントと通信します。 GDS クライアントとサーバーは、 1988 X.500 標準に基づいています。グローバル・ディレクトリー・サービスの機能は、 AIX バージョン 3.2.5 では別製品として使用可能です。 AIX バージョン 4.1 以降ではこの機能をサポートしていません。

LDAP クライアントは、ミシガン大学 3.3 ソース・コードに基づいています。LDAP クライアントは、 GDA からの LDAP API を受け付け、 LDAP プロトコルを介して LDAP サーバーと通信します。

LDAP サーバーとは、LDAP クライアントから LDAP プロトコルを受け付け、そのディレクトリーのオブジェクトにアクセスするサーバーです。 LDAP プロトコルも受け付ける X.500 サーバー、または、 LDAP プロトコルを受け付ける任意のプロプラエタリー・ディレクトリー・サービスも、LDAP サーバーになることができます。 LDAP サーバーは、 DCE では提供されないので、ユーザーが提供する必要があります。 GDA は、LDAP API を介して LDAP クライアントと通信します。

図5 は、それぞれが X.500 または LDAP サーバーを使用して他のセル内の名前にアクセスする、 2 つのセルの仮定の構成を表しています。 X.500 または LDAP サーバーに直接保管される名前は、各セルからもアクセスすることができます。 CDS は、各セル内のディレクトリー・サービスです。同じ組織が両方のセルを管理し、それらのセルは地理的位置とネットワーク・トポロジーに基づいて構成されます。

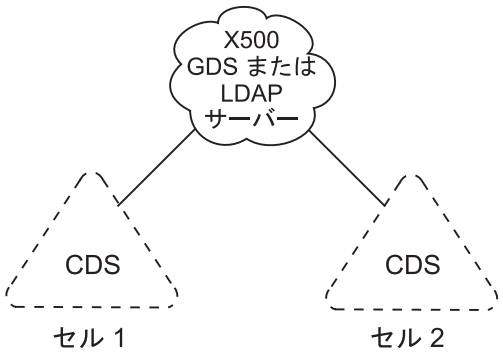


図5. セルとグローバル・ネーミング環境

GDA は DCE のコンポーネントであり、セルの相互運用を可能にします。 GDA は、いずれかのグローバル・ネーミング環境 (X.500、LDAP または DNS) を介して、CDS が別のセル内の名前にアクセスできるようにします。 GDA は独立したプロセスであり、 CDS サーバーとは別のシステム上に存在することができます。 CDS がグローバル・ネーミング環境に参加するには、少なくとも 1 つの GDA と連絡できることが必要です。

159ページの図6 は、 CDS がセル外の名前にアクセスするのを GDA で援助する方法を示しています。 CDS は、名前が自身のセルにないと判別するとその名前を GDA に渡し、 GDA はその名前に関する詳細情報を得るために適切なネーミング環境 (CDS、X.500、LDAP または DNS) を検索します。 GDA が戻す情報によって、元の CDS サーバーは、名前が置かれているセルのある CDS サーバーとコンタクトすることが可能となります。 GDA は、DNS に登録されているセルの中 (シナリオ A)、X.500 または LDAP サーバーに登録されているセルの中 (シナリオ B) で CDS が名前を検出するのを援助することができます。 GDA は、名前の構文に基づいて、どの名前サービスを使用するかを決定します。 161ページの『DCE 名のさらに詳細な分析』に、名前構文の詳細が説明されています。 GDA と、X.500、GDS または LDAP サーバーの間のインターフェースは、使用されるサーバーのタイプによって異なることに注意してください。 GDA は、 GDS クライアントとのインターフェースとして XDS/XOM API を使用します。 GDS クライアントは、 X.500 サーバーとのインターフェースに DAP プロトコルを使用します。また、GDA は、 LDAP クライアントとのインターフェースに LDAP API を使用し

ます。 LDAP クライアントは、 LDAP サーバーとのインターフェースに LDAP プロトコルを使用します。

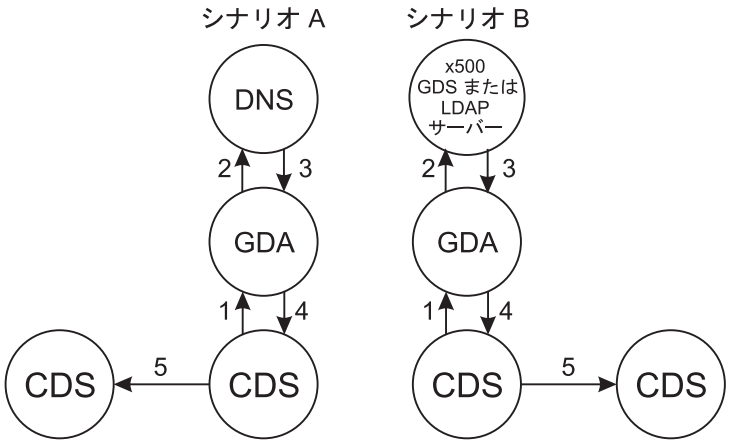


図 6. CDS、GDA、およびグローバル・ディレクトリー・サービスの対話

セルによるネーミング環境の判別の方法

セルは、ユーザーとリソースのセキュリティーおよび管理境界を明らかにするだけでなく、名前のセットの境界の判別も行います。異なるネーミング・コンポーネントがセルの内部と外部で動作しているので、セルおよびグローバル環境におけるネーミング規則も異なります。 DCE ネーミング環境では、 2 種類の名前をサポートしますが、これには、*グローバル名* と、*セル相対名* または*ローカル名* があります。以下の項では、グローバル名とローカル名の概念について紹介します。 161ページの『DCE 名のさらに詳細な分析』では、CDS および DNS の名前について詳しく説明します。

グローバル名

DCE ディレクトリー・サービスのすべてのエントリーにはグローバル名があり、このグローバル名は普遍的意味を持ち、 DCE ネーミング環境内のどこからでも使用可能です。接頭部 **/...** は、名前がグローバルであることを示します。グローバル名は、セル内のオブジェクト (CDS 内で名前が指定された) またはセルの外部のオブジェクト (X.500 内で名前が指定された) で参照することができます。

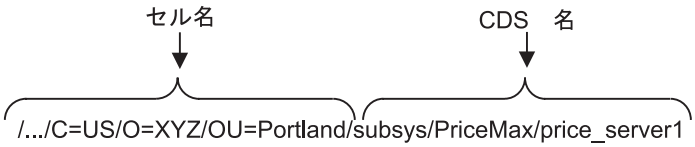
以下の例は、X.500 内で作成された項目のグローバル名を示しています。この名前は、 British 社の Widget オーガニゼーションの管理部門で働いている、 Ellie Bloggs というユーザーを表しています。

```
/.../C=GB/O=Widget/OU=Admin/CN=Ellie Bloggs
```

X.500 の名前の構文は、グローバル接頭部 **/...** と、相対識別名 (RDN) と呼ばれる 1 組のエレメントからなります。個々の RDN は、**=** (等号) によって区切られた、1 個または複数の部分のペアから構成されています。等号によって区切られた項目は複数の属性値代入 (AVA) です。 1 つのペアの最初の部分は情報のタイプを示す

省略語です。共通の省略語としては、国 (**C**)、組織 (**O**)、部門 (**OU**)、および共通名 (**CN**) があります。このペアの 2 番目の部分は値です。(X.500 の名前についての詳細は、162ページの『X.500 名』を参照してください。)

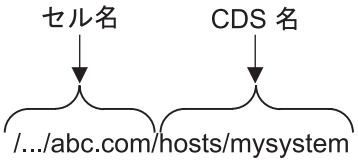
次の例は、CDS で命名された価格データベース・サーバーのグローバル名を示したものです。サーバーは、米国内のオーガニゼーションである XYZ 社の Portland 営業所によって使用されます。



この例で示すように、CDS 内で作成されたエントリーのグローバル名は、純粋の X.500 スタイルの名前とは少し違っています。名前の最初の部分である **/.../C=US/O=XYZ/OU=Portland** は、 X.500 サーバー内に存在するグローバル・セル名です。残りの部分である **/subsys/PriceMax/price_server1** は CDS 名です。

セル名が存在するのは、セルにグローバル・ネーミング環境でアクセス可能な名前を持たせる必要があるためです。 GDA は、あるセルの中の CDS が別のセルの中の名前を検索するのを援助するプロセスでセル名を調べます。セル名は DCE コンポーネントの最初の構成時に確立されます。標準的なセル間通信 (つまり、 DNS X.500 グローバル・ディレクトリー・サービスを介した通信) に関与するセルを構成する前に、 DCE 管理者は、セルを X.500 または DNS のいずれを通してアクセスする必要があるかに応じて、グローバル・ネーミング環境のいずれかから固有のセル名を入手する必要があります。

次の例は、ABC 社のホストのグローバル名を示しています。この会社のセルである **/.../abc.com** のグローバル名は、DNS の中に存在します。



スタンドアロン・セル内のセル相対名

すべての CDS エントリーには、グローバル名の他に、そのエントリーが存在するローカル・セルの中でのみ意味を持ち、そして使用可能な、セル相対名、すなわちローカル名があります。ローカル名はグローバル名の短縮形なので、ユーザー自身のセル内のリソースを参照するにはさらに便利です。ローカル名には、以下の特性があります。

- グローバル・セル名を含まない。
- ***/.:*** という接頭部で始まる。

ローカル名には、参照している名前がローカル・セル内にあることを示す ***/.:*** という接頭部があるので、グローバル・セル名は含まれていません。 CDS は、名前に ***/.:*** という接頭部を発見すると、自動的にその接頭部をローカル・セルの名前で置き換えて、グローバル名を形成します。 CDS はグローバル名とローカル名の両方を扱うことができますが、ローカル・セル内の名前を参照する場合はローカル名を使

用する方が便利です。たとえば、以下の名前は、**/.../C=US/O=XYZ/OU=Portland** という名前のセル内で使用された場合、いずれも有効です。

```
/.../C=US/O=XYZ/OU=Portland/subsys/PriceMax/price_server1
/.:/subsys/PriceMax/price_server1
```

ローカルとグローバルのディレクトリー・サービスの対話に必要なネーミング規則は、最初は分かりにくいかもしれません。ローカル・セルの外部の名前の参照が必要な環境では、以下に示す簡単な指針があれば、規則を覚えたり使用したりするのが容易になる場合があります。

- 自分のセル名を知る。
- 参照している名前が自分のセル内にあるかどうかを知る。
- 自分のセル内にある名前を使用している場合は、セル名を省略して、**/.:** という接頭部を入れることができる。
- 自分のセル外の名前を使用している場合は、そのグローバル構文を、 **/...** という接頭部とセル名を含めて入力する。
- 誰かに自分のセル内のリソースの名前を求められた場合は、 **/...** という接頭部を含むグローバル名を与える。
- 永続的ストレージ (たとえば、シェル・スクリプト内) に名前を格納する場合は、**/...** という接頭部を含む、グローバル名を使用する。ローカル名 (つまり、**/.:** という接頭部を持った名前) は対話式にのみ使用するものであり、保管はできません。 (外部セルの中からローカル名を参照すると、 **/.:** という接頭部は解決されて外部セルの名前になるため、その名前探索は失敗するか、または誤った名前になります。)

DCE 名のさらに詳細な分析

この章の残りの部分では、DCE ネームスペースを構成する各種の名前について詳しく説明します。付録 A には、 CDS 名および DNS 名の有効な文字とネーミング規則に関する詳細が記述されています。

CDS 名

すべてのセルには CDS サーバーを稼働させる少なくとも 1 つのサーバーが含まれています。 CDS サーバーは、名前の格納と保持を行い、データの作成、変更、および探索の要求を処理します。セル内の CDS サーバーが共有する名前をすべて集めた集合体を、セル・ネームスペース と呼びます。セル・ネームスペース管理者は、CDS 名を編成してディレクトリーの階層構造にすることができます。 CDS ディレクトリーは、概念的にはオペレーティング・システムのファイル・システムのディレクトリーと類似しており、管理や使用を容易にするために名前をグループ化する論理的方法です。

セル・ネームスペースでは、その下にディレクトリーを持つディレクトリーは、その下のディレクトリーの*親* であるとみなされます。その上にディレクトリーを持つディレクトリーは、その上のディレクトリーの子 であるとみなされます。セル・ネームスペースの最上位レベルは、*セル・ルート* と呼ばれます。セル・ルートを参照するには、そのセルのグローバル名か短縮形の **/.:** 接頭部を使用します。

図7 は、セル・ルートから始まる簡単なセル・ネームスペースの階層を示しています。セル・ルート (**/.:**) は、 **/.:/hosts** および **/.:/subsys** という名前のディレクトリーの親です。 **/.:/subsys** ディレクトリーは、セル・ルート・ディレクトリーの子であり、かつ **/.:/subsys/dce** ディレクトリーの親です。

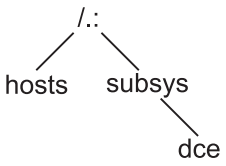


図 7. CDS ネームスペース階層の例

CDS 名の完全な指定は、左から右に向けてセル・ルートから命名されるエントリーまでを表し、*フルネーム (完全名)* と呼ばれます。フルネームの中の各要素は / (斜線) で区切られ、*単純名* と呼ばれます。たとえば、図7 に示す **/.:/hosts** ディレクトリーに、単純名が **bargle** であるホストのエントリーが含まれているとします。そのエントリーの CDS フルネームは **/.:/hosts/bargle** になります。複数の斜線が連続している場合、フルネームでは 1 個の斜線になります。

ディレクトリーが多重レベルになっていると、多数の名前の分散、アクセスの制御、および管理が容易になります。ディレクトリー階層は、名前が重複する可能性を減少させます。たとえば、 **/.:/subsys/Hypermax/printQ/server1** および **/.:/subsys/ABC/spell/server1** という名前は固有な名前です。

X.500 名

X.500 の操作は CDS の操作と似ていますが、名前の構造と探索の方法にいくつかの重要な相違があります。 CDS と同様、 X.500 と LDAP サーバーには、 X.500 の名前のアクセスと管理のためのサーバー・プロセスがあります。このプロセスを、ディレクトリー・システム・エージェント (DSA) と呼びます。また、同じグローバル・ディレクトリー・サービス実装に関係するすべての DSA の知識を結合したものを、ディレクトリー情報ベース (DIB) と呼びます。この集合知識は、多くのエントリーで構成された 1 つのグローバル・ディレクトリーとみなされます。

情報は、ディレクトリー情報ツリー (DIT) と呼ばれるルート階層の形式になって、X.500 グローバル・ディレクトリーに入っています。 DIT は CDS ネームスペースと似ています。ただし、ネームスペースと異なるところは、ネームスペースには構造や内容に関する固有の規則がないのに対して、 X.500 階層はスキーマ と呼ばれる一連の規則に影響されます。すべての X.500 DSA は、 DIB のその部分の全エントリーが従う標準スキーマを定義しなければなりません。

X.500 標準は、特定のスキーマを強制することはありませんが、主として電子メールに関する既存の X.400 標準に基づいた一般的な勧告を行います。たとえば、国や組織は DIT のルートに近いところに位置付け、人、アプリケーション、および装置の名前は階層のもっと下のところに位置付けるべきであるといったことです。X.500 には、このような勧告に従ったデフォルト・スキーマが用意されています。

どの X.500 エントリーにも識別名があり、そのエントリーを固有かつ明確に識別します。識別名は、一連の有効な相対識別名 (RDN) で構成されています。各 RDN

は、 DIT 内の特定の位置にある属性のタイプと値の 1 つまたは複数の代入で構成されています。属性型は、属性値に格納されている情報の性質を示しています。属性型と値で構成されるペアが、属性値代入 (AVA) と呼ばれるものです。 RDN に複数の AVA がある場合があります。たとえば、次のような識別名があるとします。

```
/C=us/O=osf/OU=branch1/CN=nollman,OU=doc-team
```

これは 4 個の RDN で構成されています。最後の RDN は、コンマで区切られた 2 つの AVA で構成されています。

図8 は、RDN と識別名の概念を示すとともに、これらと DIT との関係を示しています。この図は、次のものを示しています。

- スキーマによって定義された属性型の階層から成る DIT
- 属性型と値の代入から得られる RDN
- RDN の連結によって得られる識別名

X.500 名は、GDA で認識され、XDS/XOM API を介して X.500 クライアント (GDS) と、または LDAP API を介して LDAP クライアントとコンタクトして、X.500 セル名を解決します。

LDAP クライアントがコンタクトをとる LDAP サーバーは、プロプラエタリー・サーバー、または LDAP アクセス・プロトコルをサポートする X.500 サーバーでもかまいません。したがって、ユーザーは、この情報を LDAP サーバーの提供者から得る必要があります。

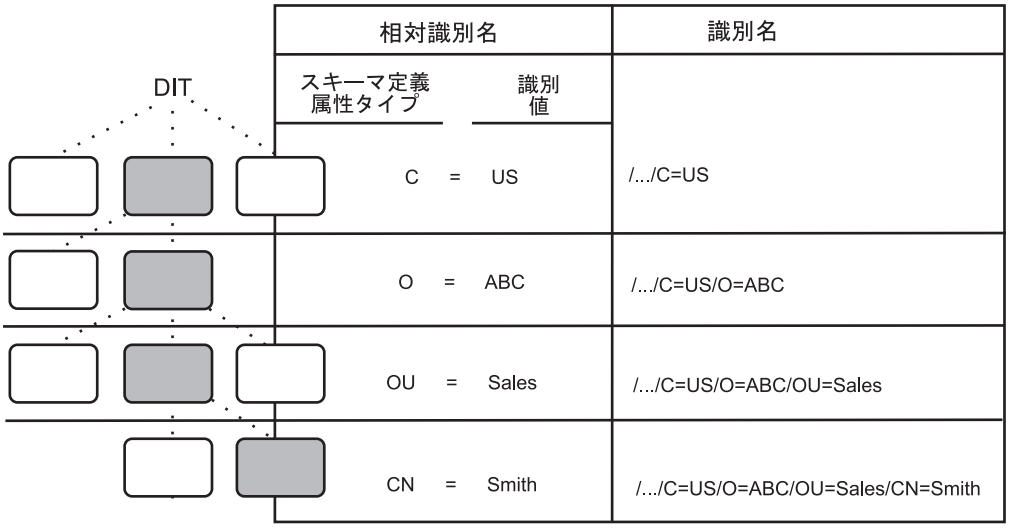


図 8. RDN と識別名

DIT の中の灰色の箱は、相対識別名という表題の欄に名前が示されているエントリーを表しています。このスキーマでは、国がルートのすぐ下で、次に組織、部門、そしてユーザー名という順に、名前を指定するように規定されています。 RDN (すなわち、識別名) を構成する各属性値のことを、*識別値* と呼びます。

図の右端の欄は DIT の各レベルのエントリーの識別名で、グローバル・ディレクトリーのルートからそのエントリーのレベルまでの RDN を連結したものです。階層

の最も下のエントリー /.../C=US/O=ABC/OU=Sales/CN=Smith は、米国の会社 ABC Company の販売部門で働いているユーザーの名前 John Smith を表しています。省略形の属性型ラベルは、国 (C)、組織 (オーガニゼーション) (O)、部門 (OU)、および共通名 (CN) を表しています。

この図はグローバル DCE の識別名に関する規則を示していることに注意してください。各識別名は、グローバル・ルートの表示 (/...) で始まります。属性型と値は等号によって分離されており、 RDN は斜線によって分離されています。 このような名前指定の規則は、 X.500 の実装システムすべてで守られているわけではありません。さらに、これらの規則は、 X.500 管理インターフェース・レベルのみで使用されます。内部では、識別名は他の方法で指定されます。

X.500 名の構造から、 X.500 と CDS との間のもう 1 つの重要な相違が分かります。 CDS 名はその属性とはまったく別のものです。つまり、これは一連のディレクトリー名で構成され、エントリーの単純名で終わっています。それに対して、 X.500 名は一連の属性型とその値のみで構成されています。

図9 は、 CDS 名と X.500 名の構造上のこうした相違を示しています。フルネームの /.:/Admin/Personnel/Employee_DB は、 Employee_DB という単純名を持つエントリーの完全なディレクトリーを指定したものです。属性とその値は CDS フルネームの一部ではありません。 X.500 の識別名 /.../C=US/O=ABC/OU=Sales は、属性型と値を、 DIT スキーマの各レベルから 1 つずつ取って連結したものです。

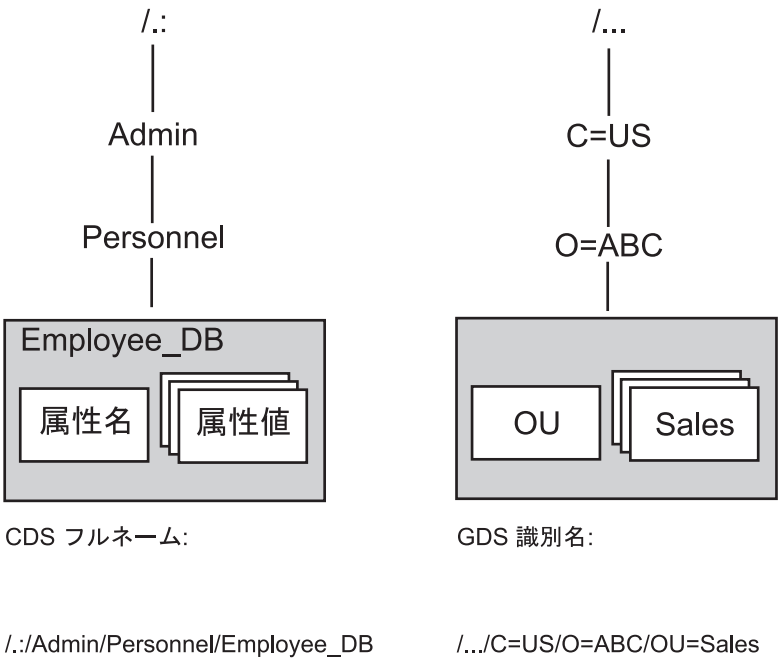


図 9. CDS 名と X.500 名の比較

注: LDAP 名 /.../OU=Sales,O=ABC,C=US は、 DCE では有効ではありません。名前は、 X.500 識別名 (.../C=US/O=ABC/OU=Sales) として指定する必要があります。

X.500 は、1 つまたは複数の属性の値を提供することによって、名前を探索する機能をサポートします。これによって、*記述名* と呼ばれるものが得られ、ある意味では、ユーザーは探している名前を記述することができます。探索機能は、価値のある機能ですが、コストも時間もかかることがあるため、 X.500 ではユーザーが探索の範囲を限定できるようになっています。探索オペレーションのサポートは、X.500 環境に限られます。

LDAP 名

LDAP 名は、X.500 名と同じ情報を含んでいますが、その構文は異なります。LDAP 名は、X.500 名の最後の RDN で始まり、 RDN の区切り記号はスラッシュ (/) ではなく、コンマ (,) です。次の例は、この相違を示しています。

```
X.500 name: /C=us/O=osf/OU=branch1/CN=nollman/OU=doc_team
LDAP name:  OU=doc_team,CN=nollman,OU=branch1,O=osf,C=us
```

DCE は X.500 セル名だけをサポートします。 GDA は、 LDAP クライアントを介して LDAP サーバーをアクセスするときに、 X.500 セル名を LDAP 構文に変換します。

DNS 名

DCE ネーミング環境では、 Internet Request for Comments (RFC) 1034 および RFC 1035 に基づく DNS のバージョンをサポートします。多くのネットワークは、ホスト名のネーム・サービスとして現在、主に DNS を使用しています。 DNS の最もよく使用されている実装例は、 Berkeley Internet Naming Domain (バークレー・インターネット・ネーム・ドメイン) (BIND) です。 BIND ネームスペースは、最上位レベルがネットワーク情報センター (NIC) の制御下にある階層ツリーです。 (ドメイン・ネームを登録するための NIC Domain Registrar へのコンタクト方法については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド － 入門* を参照してください。)

BIND ネームスペースのルートのすぐ下の名前には、 ISO 標準 3166、『Codes for the Representation of Names of Countries』 の定義に従って、 2 文字の国コード (**us** や **gb** など) が含まれます。ルートの 1 レベル下の他の名前には、 **com** (商業)、**edu** (教育)、**gov** (政府)、および **org** (その他の組織) などの、いくつかの総称管理カテゴリーが含まれています。これらの名前の所有者は、会社や組織に対して、新しい従属名を作成する認可を与えることができます。 166ページの図10 は、BIND ネームスペースのサンプル部分を示しています。 (二重引用符は、ネームスペースのルートにヌル名があり、アドレッシングできないことを示しています。) CDS 名と同様、DNS 名が入力されていないこと、すなわち、属性型と値のペアで構成されていないことに注意してください。

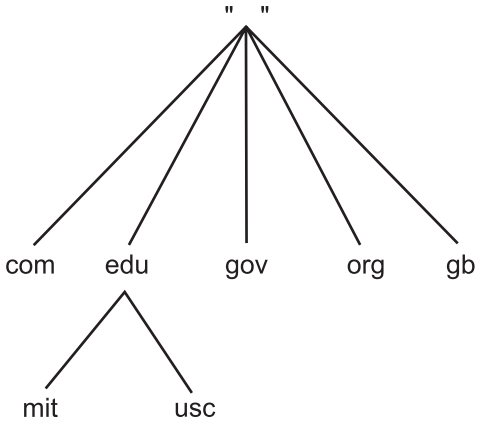


図 10. BIND ネームスペースのサンプル部分

DNS 名は、**.** (ドット) で区切られ、ネームスペースのルートから右から左に向かって配列された、階層名のストリングで構成されています。たとえば、**ai.mit.edu** という名前は、マサチューセッツ工科大学の人工知能学科の所有するネームスペースのブランチを表しています。名前の中のエレメントの順序が、 CDS 名の順序とは逆になっていることに注意してください。

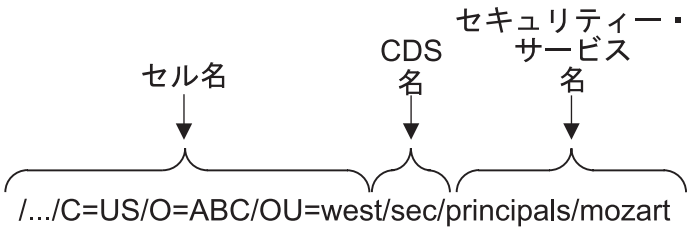
DNS セル名をグローバル DCE 名の一部として使用するには、2 つの斜線の間に DNS 名を全部指定します。たとえば、DNS 名が **ai.mit.edu** のセルに、 CDS 名が **/.:/profiles** のディレクトリーが含まれるとします。ユーザーは、このディレクトリーをグローバル名で参照するために、 **/.../ai.mit.edu/profiles** と入力する必要があります。

DCE ディレクトリー・サービスの外部の名前

すべての DCE 名が直接 DCE ディレクトリー・サービスに格納されるわけではありません。一部のサービスは、ジャンクション と呼ばれる、特殊な CDS エントリーによってセル・ネームスペースに接続します。ジャンクション・エントリーには、クライアントがディレクトリー・サービスの外部のサーバーに接続できるようにする、バインディング情報が含まれています。

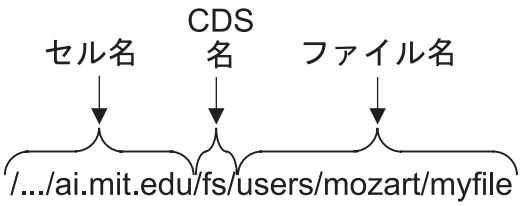
たとえば、セキュリティー・サービスは、プリンシパル (ユーザーとサーバー) のデータベース、およびそれらに関する情報 (パスワードなど) を保持します。セキュリティー・サービス・ジャンクションのデフォルト名は **/.:/sec** です。

次の例は、グローバル DCE プリンシパル名の各部分を示しています。



セル名 **/.../C=US/O=ABC/OU=west** は X.500 名です。 **sec** の部分は CDS 内のジャンクション・エントリーであり、 **principals/mozart** はセキュリティー・サービス・データベースに格納されているプリンシパル名です。

ジャンクションを使用するもう 1 つのサービスは DFS です。 DFS ファイル・セット位置サービスは、 DFS ファイル・セットをそれらが常駐するサーバーにマップする、データベースを保持します。このデータベースへのジャンクションは、**/.:/fs** というデフォルト名を持っています。次の例はグローバル DCE ファイル名のパーツを示しています。



グローバル名には、DNS セル名 **/.../ai.mit.edu** が含まれます。 **fs** の部分は CDS 内のファイル・システムのジャンクション・エントリーであり、**/users/mozart/myfile** はファイル名です。

つまり、DCE ネームスペースは、多くの異なるソースからの多くの種類の名前を接続したツリーです。ディレクトリー・サービスの GDA コンポーネントは、 X.500 または DNS などのグローバル・ネームスペースを介して、セルの外部および他のセルとの接続を提供します。これと似た方法で、ジャンクションはセル・ネームスペースから他のサービスへの下向きの接続を可能にします。

第12章 CDS の概念

セル・ディレクトリー・サービス (CDS) は、セル内部におけるリソースの命名や使用のための、整合性があり、場所に依存しない方法を提供する、高性能分散サービスです。 CDS は、CDS 名を複製する機能、すなわち、そのコピーを複数のノードに格納する機能を備えています。 CDS は複数のコピーの整合性を自動的に保ちます。複数のノードの間に名前を配布して、 1 つのノードがすべての名前を格納しなくてもすむようにすることができます。この機能は特に大きいセルにおいては価値があります。

情報の複製や配布が可能であることは、以下のようないろいろな利点があります。

- 可用性 - 複数の場所に同じ名前を格納できるので、システムやネットワークに障害が発生した場合でも、データを使用できる可能性がある。
- 効率 - 最も頻繁に使用する場所の近くに名前を格納できるので、 CDS は名前を効率よく検索できる。さらに、CDS は、一度名前を検出すると、その後のすべての検索ですぐに同じ名前に接続することができる。
- 負荷分担 - 名前が複数の場所にあるので、複数のシステムで名前の探索の負荷を分担することができる。
- 拡張性 - ネットワークが成長して CDS を使用するアプリケーションが増えても、新しい名前を簡単に収容することができる。

CDS の働き

CDS のオペレーションには、以下のいくつかの主なものが関与します。

- クライアント・アプリケーション
- サーバー
- クラーク
- クリアリングハウス

CDS はクライアント / サーバー・モデルを使用します。 CDS を使用して情報の格納と検索を行うアプリケーションは、CDS のクライアントです。クライアント・アプリケーションは、ユーザーに代わってリソースの名前を作成します。ユーザーは、クライアント・アプリケーションを介して、CDS が名前の属性として格納する、その他の情報を提供することができます。その後、クライアント・アプリケーションのユーザーがリソースをその CDS 名によって参照すると、 CDS はクライアント・アプリケーションが使用する属性のデータを検索します。

CDS サーバー・ソフトウェアを実行するシステムが CDS サーバーです。 CDS サーバーは、 CDS 名の格納と保持を行い、データの作成、変更、または探索の要求を処理します。

クラーク と呼ばれるコンポーネントは、クライアント・アプリケーションと CDS サーバーとの間のインターフェースです。どの DCE ノードも CDS クラークを実行しなければなりません。クラークはクライアント・アプリケーションからの要求を受信し、その要求をサーバーへ送信し、その結果の情報をクライアントへ戻します。このプロセスは、*検索* と呼ばれます。クラークは、クライアント・アプリケー

ションが名前の作成や変更を行う際のインターフェースでもあります。 1 つのクラークが、多くのクライアント・アプリケーションに代わって働くことができます。

クラークは、同じ情報を求めてサーバーに繰り返しアクセスする必要がないように、検索の結果をキャッシュに書き込みます (つまり、保管します)。キャッシュは、システムのリブートやアプリケーションの再始動でも情報が存続できるように、定期的にディスクに書き込まれます。 CDS アドバタイザーを停止するとクラークが停止し、キャッシュがディスクに書き込まれます。キャッシングによってパフォーマンスが向上し、ネットワーク通信量が減少します。

図11 は、 9 ノードのローカル・エリア・ネットワーク (LAN) における CDS クラークとサーバーの構成例を示しています。それぞれのノードがクラークであり、CDS サーバーは、起用された 2 つのノードで実行されます。

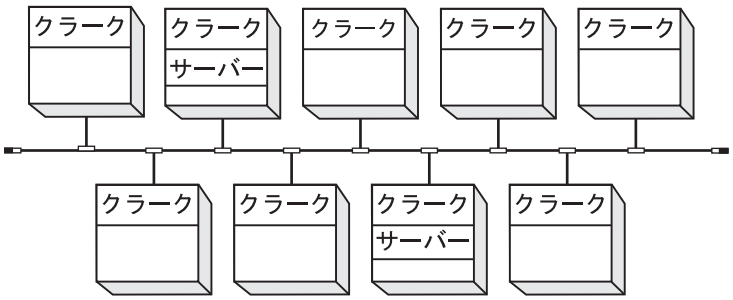


図 11. LAN における CDS クラークとサーバー

各 CDS サーバーにクリアリングハウス と呼ばれるデータベースがあり、 CDS サーバーはこの中に名前やその他の CDS データを格納します。クリアリングハウスでは、 CDS サーバーがクライアント・アプリケーションに代わってデータの追加、変更、削除、および検索を行います。 1 つのサーバー・ノードに複数のクリアリングハウスが存在してもかまいませんが、通常の構成ではお勧めできません。

171ページの図12 は、簡単な検索の際の CDS のクライアント、クラーク、サーバー、およびクリアリングハウス間の対話を示したものです。 この図では、以下の CDS 検索ステップを示しています。

1. ノード 1 のクライアント・アプリケーションがローカル・クラークに検索要求を送信します。
2. クラークはそのキャッシュをチェックして、そこに名前が見つからなければ、ノード 2 のサーバーに連絡します。
3. サーバーはそのクリアリングハウスの中に名前があるかどうかチェックします。
4. クリアリングハウスに名前が存在していたので、サーバーは要求された情報を入手します。
5. サーバーはノード 1 のクラークに情報を戻します。
6. クラークは要求されたデータをクライアント・アプリケーションに渡します。クラークは情報をキャッシュへも書き込み、次にクライアントがその同じ名前の検索を要求した時に、サーバーにコンタクトする必要がないようにします。

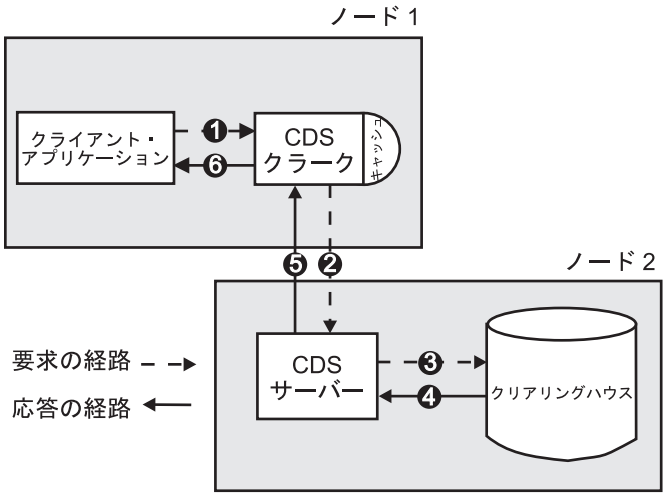


図 12. CDS 検索のサンプル

レプリカとその内容

ディレクトリーは、セルのネームスペース全体に名前を配布し複製するための単位です。ディレクトリーの各物理コピーは、オリジナルを含めて、*レプリカ* と呼ばれます。ディレクトリーのレプリカを作成する時は、そのディレクトリー内のエントリーもすべて複製します。

レプリカはクリアリングハウスに格納されます。 クリアリングハウスは、ある特定のサーバーにあるディレクトリー・レプリカの集合体と考えることができます。 1 つのクリアリングハウスにディレクトリーを作成した後で別のクリアリングハウスにもそのレプリカを作成しておけば、情報の検索の可用性を高めることができます。 CDS は、ディレクトリーのすべてのレプリカの内容の整合性が保たれるよう、定期的に確認を行います。

以下の 2 つのタイプのレプリカがあります。

- マスター
- 読み取り専用

レプリカのタイプは、そのレプリカに対して行うことができる処理および、CDS がそれを更新する方法に影響を与えます。 CDS がデータの検索や変更を行う際に使用するレプリカのタイプはユーザーには分かりません。ただし、2 つのタイプの違いを理解しておくことは役立ちます。

マスター・レプリカは、セルのネームスペースにおける特定ディレクトリーの最初のインスタンスです。必要な場合は、ディレクトリーのコピーを作成した後で、異なるレプリカをマスターに指定することができます。ただし、それぞれのディレクトリーのマスター・レプリカは一時点で 1 つしか存在できません。 (ディレクトリーのマスター・レプリカの再指定の方法についての詳細は、 237ページの『第21章 ネームスペースの再構成』を参照してください。)

マスター・レプリカは、ディレクトリーのレプリカの中で唯一、直接変更可能なレプリカです。 CDS はマスター・レプリカの中の情報を作成、変更、および削除することができます。マスター・レプリカは修正可能なので、読み取り専用レプリカより多くのオーバーヘッドが必要になるため、CDS は、マスター・レプリカに加えられた変更を定期的に更新します。

読み取り専用レプリカは、情報の検索のみに利用できるディレクトリーのコピーです。 CDS は、読み取り専用レプリカの中の名前については作成、変更、または削除を行わず、マスター・レプリカに対して行った変更内容に合わせて更新するだけです。

レプリカには、以下の 3 種類のエントリーを含めることができます。

- オブジェクト・エントリー
- ソフト・リンク
- 子ポインター

CDS 優先クリアリングハウス拡張

この拡張によって、 CDS 情報を得るためにクライアントがクリアリングハウスにアクセスする順序でそれらのクリアリングハウスがランキングされるため、 CDS クライアントのパフォーマンスは向上します。これは、 CDS サーバーとの関係における CDS クライアントの位置に関連付けられたデフォルトを使用することで自動的に行うことも、あるいはセル管理者による手動の変更操作で行うこともできます。

この拡張は、たとえば、パフォーマンスの良好な複数の LAN がパフォーマンスの悪い 1 つの WAN に接続されており、そのおのおのの LAN のクリアリングハウス中に CDS レプリカがあるといった状況で有効です。この機能においては、ランキングが最上になるクリアリングハウスはクライアントのあるマシン上のものであり、そのクライアントと同じ LAN にあるクリアリングハウスが次にきます。ローカルのクリアリングハウスは、別の場所にあるクリアリングハウスに比べて優先順位が高くなります。クライアントが別の場所にあるクリアリングハウスを使用するのは、ローカルのクリアリングハウスが要求を満たすことができない場合に限られます。管理者はデフォルトを変更して、クリアリングハウスとの通信の順序をより具体的に指定することができます。優先順位の指定は、それぞれのクリアリングハウスにランクを割り当てることによって行います。詳しくは、208ページの『クラークと特定クリアリングハウスとの通信の設定』 を参照してください。

オブジェクト・エントリー

オブジェクトは、CDS 名が与えられた実リソース (ディスク、アプリケーション、あるいはノードなど) です。オブジェクト名が作成されると、クライアント・アプリケーションと CDS ソフトウェアはその名前と一緒に格納すべき属性を提供します。属性は、属性名と値 (複数の場合があります) で構成され、それはオブジェクトの特定の操作属性を記述します。名前とその属性によって、オブジェクト・エントリー が構成されます。クライアント・アプリケーションが名前の検索を要求すると、 CDS は関連する属性の値を戻します。

オブジェクト・エントリーは一般に、クライアント・アプリケーション・インターフェースを介して作成され、管理されます。たとえば、 DCE 制御プログラムと

RPC ランタイムのネーム・サービス・インターフェース (NSI) を使用すれば、ユーザーは、RPC サーバー、グループ、およびプロファイルを表すエントリーを作成することができます。これらは特殊なエントリーであり、これによって RPC アプリケーションはサーバーを見つけたり選択したりすることが可能となります。 (この目的で RPC が CDS をどのように使用するかについての詳細は、 *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide* を参照してください。)

DCE 制御プログラム (**dcecp**) を介してオブジェクト・エントリーを作成することもできます。 (**dcecp** を使用してオブジェクト・エントリーを作成および管理するコマンドの詳細については、この資料の 1ページの『第1部 DCE 制御プログラム』および *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

各オブジェクトは定義クラスを持つことができますが、これはオブジェクト・エントリーのオプション属性です。ディレクトリー・サービスを使用する DCE コンポーネントは、自身のオブジェクト・クラスを定義し、ディレクトリー・サービスが代わって格納できるようにするためのクラス固有属性を提供することができます。クラス固有属性は、それが関連する特定クラスのオブジェクトのみに意味を持ちます。

クリアリングハウス・オブジェクト・エントリーは、 CDS によってあらかじめ定義された、特殊クラスのオブジェクトを表します。クリアリングハウス・オブジェクト・エントリーは、ネットワーク内のクリアリングハウスの位置を示すポインターの役割をします。 CDS は、クリアリングハウス内のデータの検索や更新のために、このポインターを必要とします。

ユーザーがクリアリングハウスを作成すると、 CDS は自動的にそのクリアリングハウス・オブジェクト・エントリーを作成します。クリアリングハウス・オブジェクト・エントリーには、クリアリングハウスと同じ名前が付けられます。クリアリングハウス・オブジェクト・エントリーは、実際のリソースを記述するという点では他のオブジェクトと似ていますが、 CDS による内部使用専用であるという点が異なります。クリアリングハウスは、セルのルート・ディレクトリーにのみ作成することができます。したがって、クリアリングハウス・オブジェクト・エントリーはすべて、セルのルート・ディレクトリーに格納されます。 CDS 自体は、必要に応じて、クリアリングハウス・オブジェクト・エントリーを更新し管理します。これは、まれに問題解決の状態で必要になる以外、いかなる外部管理も必要としません。 (このような状態の場合は、ベンダーに連絡して援助を求めてください。)

ソフト・リンク

ソフト・リンク は、セル・ネームスペース内のオブジェクト・エントリー、ディレクトリー、あるいは他のソフト・リンクに対して代替名を提供する、ポインターです。既存の名前から新しい名前をポイントするソフト・リンクを作成することによって、セルのネームスペースの小規模な再構成を行うことができます。ソフト・リンクは、何かに複数の名前を指定して、いろいろなユーザーが自分にとって最も意味のある方法で名前を参照できるようにする方法でもあります。

ソフト・リンクは永続的なものにすることもできれば、指定期間が過ぎたら期限切れにすることもできます。ソフト・リンクがポイントする名前が削除されると、CDS は、このソフト・リンクの有効期限が切れたときに自動的にそれを削除します。

CDS 管理者は、ソフト・リンクを慎重に使用する必要があります。セルのネームスペースを完全に設計し直すためや、オブジェクト・エントリーのフルネームを使用したくないユーザーに近道を提供するためなどには、ソフト・リンクを使用しないでください。ソフト・リンクを濫用すると、CDS 名の追跡と管理が困難になります。

子ポインター

子ポインター は、セルに対して以下の種類の接続を提供します。

- セルのネームスペース内のディレクトリーとそのすぐ下にある別のディレクトリーとの間の接続

ユーザーとアプリケーションは子ポインターを作成しません。誰かが新しいディレクトリーを作成すると、CDS は自動的に子ポインターを作成します。子ポインターは、それがポイントするディレクトリーの親 (1 レベル上) のディレクトリーの中に作成されます。 CDS は、名前を検索しようとするときに、ディレクトリー・レプリカを見つけるために子ポインターを使用します。まれな問題解決の状態の場合を除き、子ポインターを管理する必要はありません。

まとめ

要約すると、セルは、1 つのセル内の 1 つまたは複数の CDS サーバーによって共有され管理される、名前の完全なセットで構成されています。名前でディレクトリー、オブジェクト・エントリー、ソフト・リンク、または子ポインターを指定することができます。セルのネームスペースを論理的に表すと、ディレクトリーとそれに含まれている名前の階層構造になります。ディレクトリーの各物理インスタンスを*レプリカ* と呼びます。名前はレプリカに物理的に格納され、レプリカはクリアリングハウスに格納されます。クリアリングハウスが入っていて、 CDS サーバー・ソフトウェアを実行するノードはすべて、*CDS サーバー* です。

175ページの図13 は、CDS サーバー・ノードのコンポーネントを示したものです。すべてのサーバーが、ディレクトリー・レプリカを含むクリアリングハウスを少なくとも 1 つ管理しています。レプリカにはオブジェクト・エントリー、ソフト・リンク、および子ポインターを含めることができます。この図は、 1 つのレプリカと、レプリカ内に入れることができるエントリーを各タイプについて 1 つ示しています。通常、1 つのクリアリングハウスには多くのレプリカが含まれており、 1 つのレプリカには多くのエントリーが含まれています。

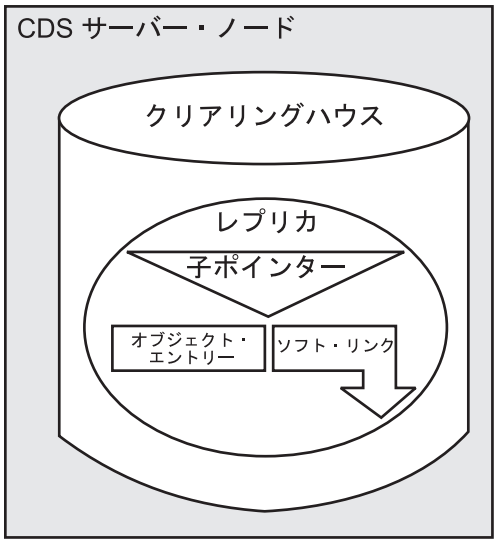


図 13. CDS サーバー・ノードのコンポーネント

セル・ディレクトリー環境におけるセキュリティー

保護 DCE セル操作では、ユーザーの識別が DCE 認証サービスによって検証されるまで、サーバーはユーザーの要求を完了しません。そのため、たとえば、CDS サーバーは識別が検証された場合にのみ、ユーザーに新しいディレクトリーの作成を許可します。ユーザーが主張するとおりの人物であるかをどうかを検証するプロセスを、*認証* と呼びます。証明は、ユーザー名 (またはプリンシパル名) を特殊なパスワードと組み合わせた形式になります。

CDS サーバー自体は、次の 2 つの理由により、認証されたプリンシパルでなければなりません。

- クライアントに対して自身が信頼できるものであることを証明するため
- 共用するデータの変更や管理の許可を受けていることを互いに証明するため

CDS サーバーのプリンシパル名は、構成プログラムによって自動的に選択され、セル内のすべての CDS サーバーの名前を含むグループに入れられます。このグループは、 DCE セキュリティー・サービス・データベース内のエントリーとして格納されます。 CDS サーバーと最初にコンタクトした後、クラークは、DCE セキュリティー・サービスを介して、そのサーバーがメンバー・グループの有効なメンバーであることを確認します。

認証は、それ自体が目的ではなく、許可のプロセスにおける 1 つのステップです。プリンシパルの識別を検証した後、ソフトウェアは次に、そのプリンシパルが要求されたアクションの実行に必要な許可を受けているかどうかを判別しなければなりません。これを*認可* と呼びます。したがって、新しいディレクトリーを作成するためには、前の例のユーザーは、認証されるだけでなく、適切な許可も持っていなければなりません。

サーバーは、複製されたデータを共用し、かつ変更するので、互いに認証し合う必要があります。たとえば、サーバー A とサーバー B が両方とも同じディレクトリーのレプリカを格納するとします。各ディレクトリーには、そのディレクトリーを保守する許可を受けた全サーバーのリストが関連付けられています。ユーザーがサ

ーバー B にあるレプリカ内のエントリーを変更する場合、サーバー B はサーバー A にこの変更を通知をしなければなりません。サーバー B が認証を受けたプリンシパルであり、そのディレクトリーの変更の許可を受けたプリンシパルの 1 人でない限り、サーバー A はこの更新を受け入れません。

CDS の許可には読み取り、書き込み、挿入、削除、テスト、制御、および管理があります。それぞれの許可は、関連付けられている名前に応じて意味が少し異なりますが、一般に次のような意味を持っています。

- 読み取り許可は、ユーザーによるデータの表示を可能にします。
- 書き込み許可は、ユーザーによるデータの追加または変更を可能にします。
- 挿入許可は、ユーザーによるディレクトリーでのエントリーの作成を可能にします。
- 削除許可は、ユーザーによるエントリーの削除を可能にします。
- テスト許可は、ユーザーが、名前の属性に特定の値があるかどうかを、値を見ることができなくても (つまり、名前の読み取り許可がなくても) テストすることができるようにします。この許可の主な利点は、値をチェックするより便利な方法をアプリケーション・プログラマーに提供するという点です。この方法では、アプリケーションは、値のセット全体を読むことなく、特定の値についてテストすることができます。
- 制御許可は、ユーザーによるエントリーのアクセス制御リスト (ACL) の管理を可能にします。
- 管理許可は、ユーザーによるディレクトリー・レプリカの管理を可能にします。

認証できないユーザー、または認証を受けていないオペレーションを意図的に要求するユーザーに対しては、特殊な ACL を定義できることに注意してください。このような場合、ユーザーの識別は検証されないため、認証を受けていないユーザーの ACL エントリーが、要求したアクションを実行する許可をそのユーザーが受けているかどうかを判別します。 (認証されていないユーザーに対する ACL の作成の詳細については、このガイドの 315ページの『第6部 DCE セキュリティー・サービス』を参照してください。)

CDS のユーザー・インターフェース

CDS にはいくつかのエンティティー があり、これらは、DCE に用意されているユーザー・インターフェースを介して管理することができます。 CDS エンティティーとは、CDS ソフトウェアの個別に管理できる部分をいいます。 CDS のディレクトリー、ソフト・リンク、およびオブジェクト・エントリーは、 DCE ユーザー・インターフェースによって管理する、最も一般的なエンティティーです。ただし、一部のオブジェクト・エントリーは、通常、それを作成したクライアント・アプリケーションによって管理されます。

DCE 制御プログラムは、 CDS エンティティーを管理するための多くのコマンドを提供します。 このガイドの 191ページの『第15章 DCE ディレクトリー・サービスの管理』に、これらのコマンドについての情報が記載されています。

CDS 名のアクセスと管理を可能にする、他の DCE ユーザー・インターフェースもあります。たとえば、DCE セキュリティー・サービスと一緒に提供される **dcecp acl** オブジェクトなどの ACL エディターを使用して、ユーザーは CDS ディレク

トリーおよびその内容へのアクセスを制御することができます。 RPC アプリケーション・プログラマーは、**dcecp** を使用して、セルのネームスペース内にサーバー・エントリー、グループ、および構成プロファイルを作成することができます。

第13章 CDS による名前の探索方法

この章では、名前とそれが記述する物理リソースとの間の関係を示すとともに、CDS が名前の検索要求を処理する方法について説明します。これらの概念を理解すると、セル・ネームスペース内におけるクリアリングハウスやディレクトリーの場所を計画するのに役立ちます。また検索エラーや障害などを検出した場合に、問題の発生源を分離するのにも役立ちます。この章の図は、一般的な DCE セル・ネームスペースの実際の構造を反映したものではないことに注意してください。簡略化のために、この図ではディレクトリーやディレクトリーのレベル数を少なく示してあります。

名前からリソースへの変換

論理ネームスペース階層内のディレクトリー名がクリアリングハウス内の物理レプリカに変換されるのと同様、 CDS 名は、CDS が内部で使用するかあるいはクライアント・アプリケーションが使用する、物理リソースに変換されます。名前の属性により変換が可能になります。この節では、 CDS 名とそれが記述する物理リソースとの関係について説明します。

180ページの図14 は、論理ネームスペースにおける 3 つのディレクトリーとその内容、およびそのディレクトリー・レプリカを 2 つのクリアリングハウスに物理的に実装する方法を示しています。クリアリングハウスそのものは、ノード 1 では **/.:/Paris_CH**、ノード 2 では **/.:/NY_CH** という CDS 名を持ちます。 **_CH** という接尾辞は、推奨されるクリアリングハウスの命名規則です。 **/.:/Paris_CH** クリアリングハウスには、ルート・ディレクトリーと **/.:/subsys/PrintQ** ディレクトリーのレプリカが含まれます。 **/.:/NY_CH** クリアリングハウスには、ルート・ディレクトリーと **/.:/subsys** ディレクトリーのレプリカが含まれます。すべてのディレクトリーのレプリカを最低 2 つずつ作成することを習慣にしておくことをお勧めします。このため、**/.:/subsys** ディレクトリーと **/.:/subsys/PrintQ** ディレクトリーは、それぞれ、セル内のどこかで、少なくとも 1 つの他のクリアリングハウスでレプリカを作成する必要があります。

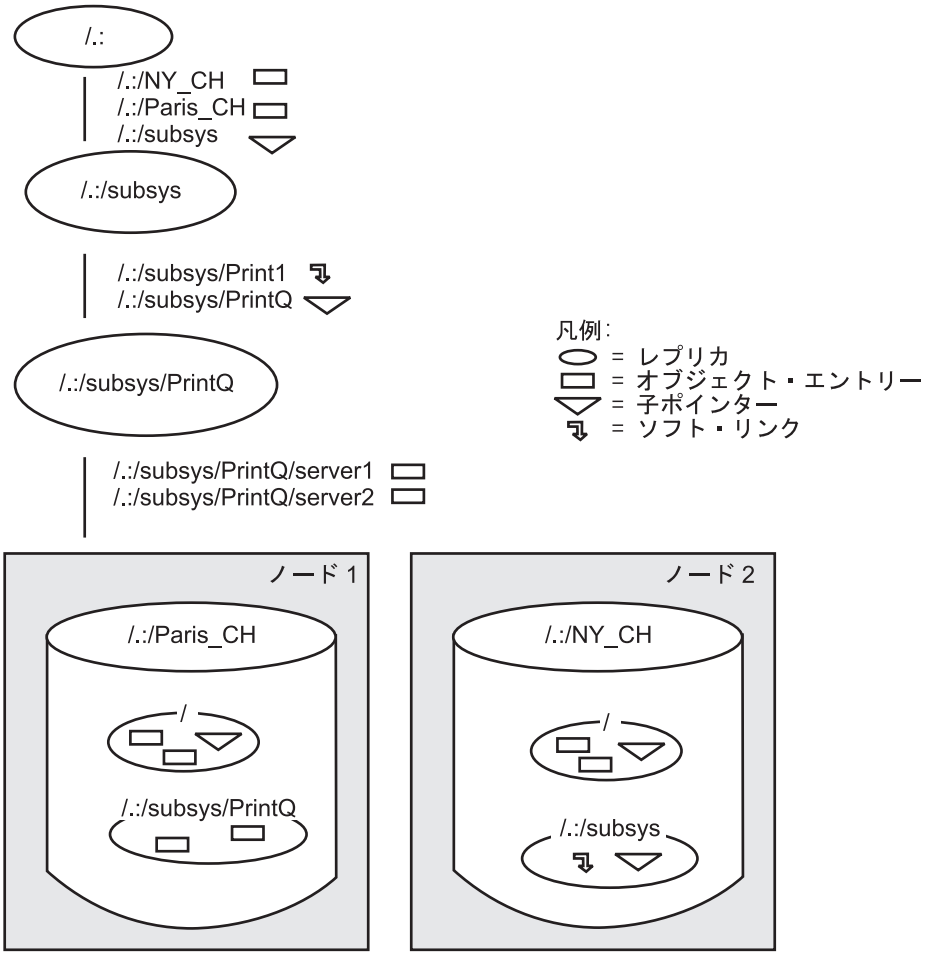


図 14. ネームスペースの論理表示と物理表示

リソースの物理的位置を見つけるために、CDS はその名前と関連する属性を検索します。 181ページの図15 から 183ページの図17 は、各種の CDS 名とそれらが記述するリソースとの間の接続を示しています。これらの図は、図14 のネームスペースに基づいています。 181ページの図15 から 183ページの図17 の中のすべての名前は、セル・ルートを表す **/.:** という接頭部が使用されていることから明らかなように、同一のセル・ネームスペース内にあります。 (複数のセルにまたがる名前の解決については、 251ページの『第22章 セル間ネーミングの管理』を参照してください。)

181ページの図15 は、2 つのクリアリングハウス・オブジェクト・エントリーと、それらが記述しているクリアリングハウスとの間の関連を示しています。クリアリングハウス・オブジェクト・エントリーは、クライアント・アプリケーションではなく CDS ソフトウェアが作成し、使用し、保守するという点で、他の種類のオブジェクト・エントリーと異なっています。ただし、ネットワーク内の物理リソース、つまり、クリアリングハウス、を記述するという点は他のすべてのオブジェクト・エントリーと似ています。 CDS は、ユーザーがクリアリングハウスを作成して名前を付けたときに、自動的にオブジェクト・エントリーを作成します。

181ページの図15 は、ノード 1 の **/.:/Paris_CH** という名前のクリアリングハウスをポイントする **/.:/Paris_CH** と、ノード 2 の **/.:/NY_CH** という名前のクリアリン

グハウスをポイントする **/.:/NY_CH** という、 2 つのクリアリングハウス・オブジェクト・エントリーを示したものです。各クリアリングハウス・オブジェクト・エントリーには **CDS_CHLastAddress** 属性と呼ばれる属性があり、その **Tower** 副属性には、クリアリングハウスが常駐するノードに CDS がコンタクトするために使用する、 RPC バインディング情報が含まれています。 (CDS 属性とその説明については、付録 B を参照してください。)

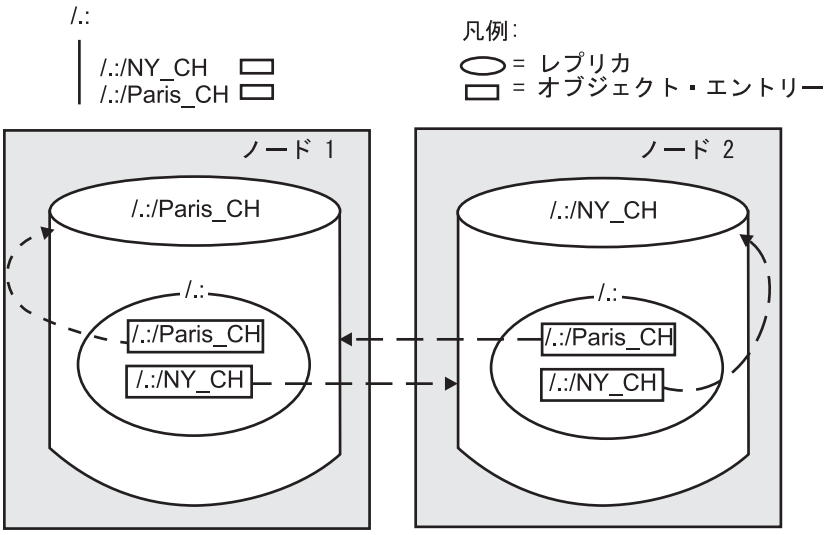


図 15. クリアリングハウス・オブジェクト・エントリーとクリアリングハウス

182ページの図16 は、ソフト・リンク、それがポイントするオブジェクト・エントリー、およびオブジェクト・エントリーが記述するリソースとの間の関係を示しています。ソフト・リンク **/.:/subsys/Print1** には **CDS_LinkTarget** と呼ばれる属性があり、これには、そのリンクがポイントする **/.:/subsys/PrintQ/server1** という名前のオブジェクト・エントリーが含まれています。このオブジェクト・エントリーは、 **PrintQ** と呼ばれるアプリケーションで使用される、プリント・サーバー・マシンを記述するものです。 **/.:/subsys/PrintQ/server1** オブジェクト・エントリーを含むレプリカは、 **/.:/Paris_CH** クリアリングハウスに存在します。このオブジェクト・エントリーには **CDS_Towers** と呼ばれる属性があり、その **Tower** 副属性には、 **PrintQ** アプリケーションがプリント・サーバー・マシンとコンタクトできるようにする、 RPC バインディング情報が含まれています。

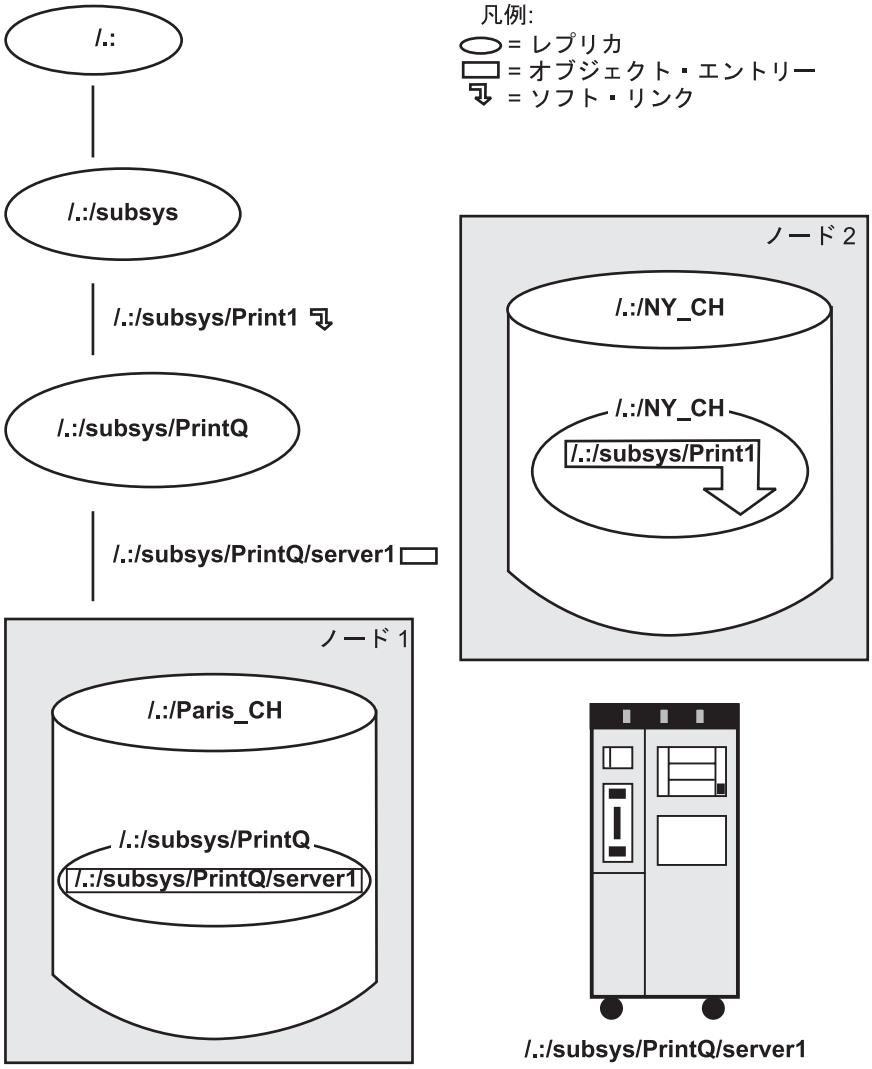


図 16. ソフト・リンクとその解決

183ページの図17 は、ディレクトリーとそれに関連する子ポインターとの間の関係を示したものです。この図では、子ポインターはそれと関連したディレクトリーと同じ名前を持っていますが、子ポインターは、ネームスペース内の別のエントリーであり、子ポインターはそれが参照するディレクトリーの親に存在していることを示しています。

両方のクリアリングハウスのルート・レプリカには、 **/.:/subsys** ディレクトリーの子ポインターが含まれています。 **/.:/subsys** 子ポインターには **CDS_Replicas** と呼ばれる属性があり、これには、 **/.:/subsys** ディレクトリーが存在する **/.:/NY_CH** クリアリングハウスの名前とアドレスが含まれています。

/.:/NY_CH クリアリングハウスでは、 **/.:/subsys** ディレクトリーのレプリカには、**/.:/subsys/PrintQ** ディレクトリーの子ポインターが含まれています。この子ポインターの **CDS_Replicas** 属性には、 **/.:/subsys/PrintQ** ディレクトリーのレプリカが存在する、 **/.:/Paris_CH** クリアリングハウスの名前とアドレスが含まれています。

ディレクトリーに複数のレプリカがある場合 (こういったケースが普通ですが)、**CDS_Replicas** 属性にディレクトリー・レプリカを含むすべてのクリアリングハウスがリストされます。 **-replica** オプションと **-clearinghouse** オプションを指定した **dcecp directory show** コマンドを使用して、この属性を表示することができます。

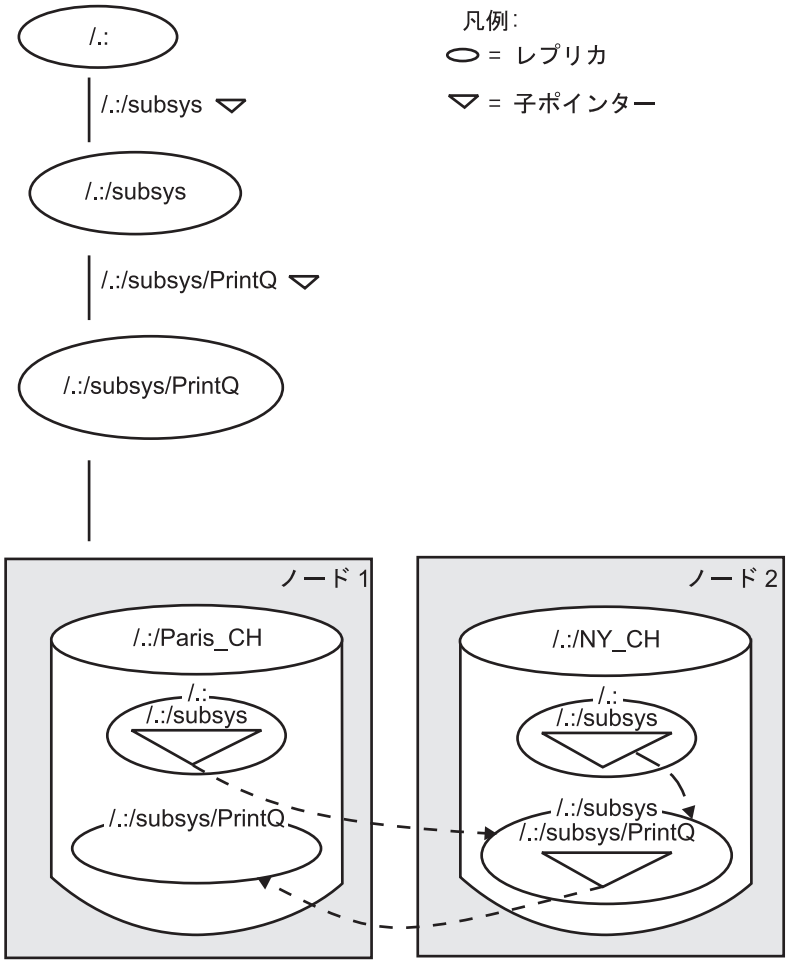


図 17. 子ポインターとディレクトリー

CDS による名前の検索方法

180ページの図14 から図17 で示したように、 CDS は、リソースの物理的な場所の情報を、その名前と関連した 1 つまたは複数の属性を検索することによって見つけます。ただし、まず最初に、クラークは、名前を見つける方法を知っていなければなりません。クラークのキャッシュにまだ名前が存在しない場合は、クラークは、名前を検索するときにコンタクトすべき CDS サーバーを少なくとも 1 つは知っていなければなりません。

クラークは、CDS サーバーとその位置を、以下の 3 つの方法のいずれかで知ることができます。

- 送信請求と通知プロトコル
- 定期の検索時

- **dcecp cdscache create** コマンド

送信請求と通知プロトコル

同一の LAN 上にあるクラークとサーバーは、送信請求と通知プロトコルを使用して通信します。サーバーはその存在を LAN 上のクラークに通知するために、一定間隔でメッセージをブロードキャスト (ブロードキャスト) します。通知メッセージにはサーバーが属するセル、サーバーのネットワーク・アドレス、およびそれが管理するクリアリングハウスに関するデータが含まれています。クラークは、LAN 上のこうした通知を listen することによって、サーバーを確認します。また、クラークは、通知を要求する送信請求メッセージを始動時に送信します。

検索

検索時に、クラークが探している名前がクリアリングハウスにないと、そのクリアリングハウスを管理しているサーバーは、その名前を検索すべき他の場所に関するできる限りの情報をクラークに提供します。検索しているフルネームの一部であるレプリカがクリアリングハウスに含まれているが、目標の単純名を含むレプリカが含まれていない場合、クリアリングハウスは自身が持っているレプリカの適切な子ポインターからのデータを戻します。このデータは、クラークが目標の単純名へのパスの中で次の子ディレクトリーを検索する時に役立ちます。子ポインターの **CDS_Replicas** 属性には、クリアリングハウス名とバインディング情報の形式で、このデータが含まれています。

dcecp cdscache create コマンド

DCE 管理者は、**dcecp cdscache create** コマンドを実行して、クラークのキャッシュの中にサーバーに関する知識を作成することができます。このコマンドは、サーバーとクラークの間が広域ネットワーク (WAN) によって隔てられており、そのためにクラークが LAN 上の通知からはサーバーについて知ることができない場合に役立ちます。

185ページの図18 は、オブジェクト・エントリーを探し出す時に、クラークがセル・ネームスペースのルートから下向きに処理して行く方法の例を示しています。オブジェクト・エントリー **/.:/Sales/Spell** は、会社のロンドン販売本部にあるスペル・チェック・サーバーを記述しています。

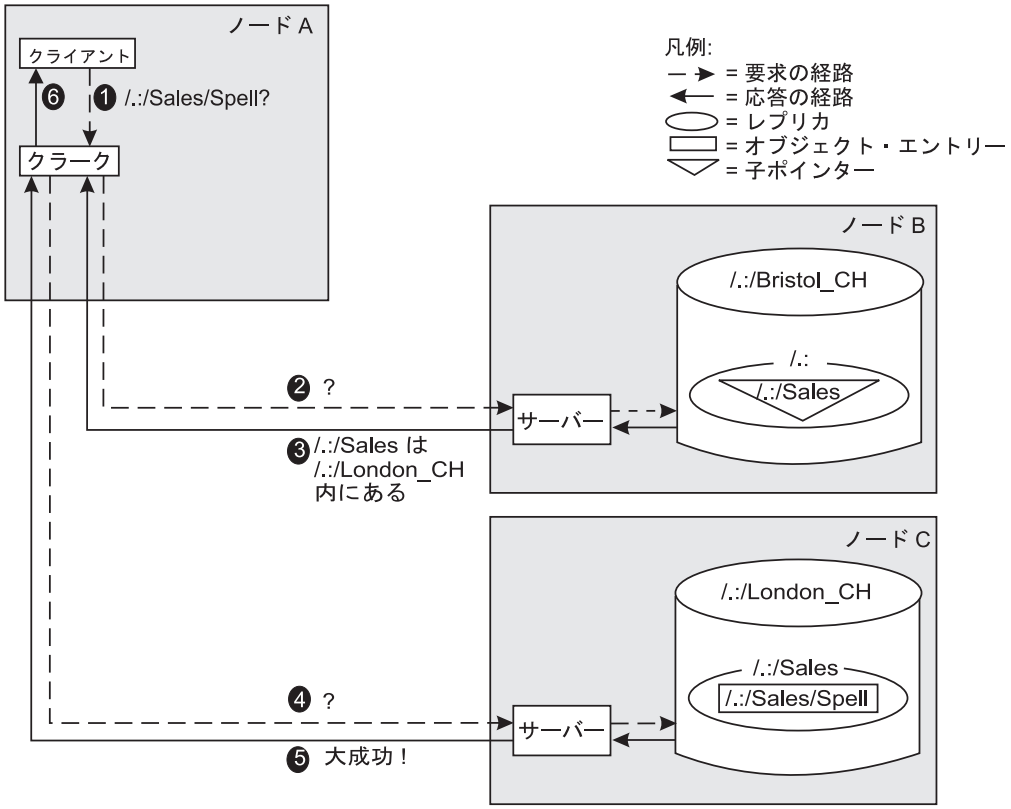


図 18. クラークによる名前の検索方法

図18 に示されているように、クラークは、以下のステップを実行することによって、必要なオブジェクト・エントリーを探し出します。

1. ノード A において、スペル・チェック・アプリケーションが **/.:/Sales/Spell** サーバーのネットワーク・アドレスを要求します。クラークのキャッシュにはその名前がなく、かつクラークがこれまでに知っているクリアリングハウスは、ノード B の **/.:/Bristol_CH** だけです。
2. クラークはノード B のサーバーに検索要求でもってコンタクトします。
3. **/.:/Bristol_CH** クリアリングハウスには目標のオブジェクト・エントリーはありませんが、ルート・ディレクトリーのレプリカは含まれています。ルートの中の **/.:/Sales** 子ポインターから、クラークは、 **/.:/Sales** ディレクトリーのレプリカを持っているクリアリングハウスにコンタクトする方法を知ることができます。ノード B のサーバーは、このデータをクラークに戻して、 **/.:/Sales** のレプリカがノード C の **/.:/London_CH** クリアリングハウスにあることを知らせます。
4. クラークは、ノード C 上のサーバーに、検索要求でもってコンタクトします。
5. ノード C のクリアリングハウスの中の **/.:/Sales** レプリカには **/.:/Sales/Spell** オブジェクト・エントリーが含まれているので、サーバーはスペル・チェック・サーバーのアドレスをクラークに渡します。
6. クラークはこの情報をクライアント・アプリケーションに戻し、クライアント・アプリケーションはこれを使用してスペル・チェック・サーバーへのリモート・コールを行うことができます。

185ページの図18 に示したような長い検索は、クラークがキャッシュを設定し、クリアリングハウスやその内容をよく知るようになれば、通常はあまり起こりません。ただし、この図には初期の検索に関係する可能性のある、リソースと接続が示されています。またこの図には、ネームスペース内の親ディレクトリーと子ディレクトリーとの間の接続を維持しておくことの重要性も示されています。もしどこかでディレクトリー・パスが切断されたり、クリアリングハウスに到達できなくなったりした場合、クラークは名前を見つけることができなくなります。

第14章 CDS によるデータの更新方法

ネームスペースに名前が存在していれば、適切なアクセス権を持つユーザーは、その名前に関連したデータを変更することができます。 CDS データの追加、修正、あるいは削除は、最初は 1 つのレプリカ、つまりマスター・レプリカのみに行われます。この章では、 CDS が他のレプリカの一貫性を保つための主な方法である、*更新伝搬* および*スカルク・オペレーション*について紹介します。また CDS データの一貫性の確保に役立つ 2 つのタイム・スタンプについても説明します。このトピックで説明する概念を理解すれば、ディレクトリーの内容と複製をより効率よく計画することができます。

更新伝搬

更新伝搬とは、 1 つの変更が行われたばかりのディレクトリーの全レプリカにその変更を直ちに適用しようとする試みのことです。この主な利点は、それによってそれぞれの変更が効果的でタイムリーな方法で送達されることです。

ただし、スカルク・オペレーションの場合とは異なり、更新伝搬にはすべてのレプリカで即時に変更が行われるという保証はありません。特定のレプリカが使用不能の場合、更新伝搬は失敗するわけではなく、そのレプリカでは変更が後で行われます。スカルク操作の場合は、レプリカが再度利用可能になったときに、そのセット内の他のレプリカと整合性を保つことが保証されます。

CDS_Convergence と呼ばれるディレクトリー属性を調整することによって、CDS が更新伝搬を試みる際に使用する持続性の程度を調整することができます。コンバージェンスはディレクトリーのスカルクの頻度にも影響します。 (ディレクトリーのコンバージェンスの表示および変更の詳細については、 215ページの『第18章 CDS ディレクトリーの管理』を参照してください。)

スカルク操作

スカルク操作とは、更新の集合を定期的に分配する操作のことです。その主な機能は、更新伝搬時に届けられなかった可能性のある変更内容をレプリカが確実に受け取ることができるようにすること、および古い情報をネームスペースから削除することです。

スカルク保守機能には次のものが含まれます。

- 期限切れのソフト・リンクの削除。ソフト・リンクの作成時に有効期限を指定することができます。
- 子ポインターの保守。これには削除されたディレクトリーへのポインターの削除も含まれます。
- 削除されたレプリカに関する情報の削除。

CDS は各ディレクトリーを個別にスカルクします。スカルクの際に、CDS は、最後の正常なスカルク以後にマスター・レプリカに加えられた変更をすべて収集しておいてから、その変更内容をディレクトリーのすべての読み取り専用レプリカに分配します。成功と見なすには、全レプリカがスカルクで利用できなければなりませ

ん。 CDS は、レプリカにコンタクトできない場合、コンタクトできるレプリカ内の変更を続ける一方で、更新できなかったレプリカを通知するイベントを生成します。 CDS はその後、正常に完了するまで、定期的にスカルクの試みを繰り返します。

スカルクは、次の 3 つの方法のいずれかで開始できます。

- CDS マネージャーは、コマンドを入力して、ディレクトリー上で即時スカルクを開始することができます。
- CDS は、他のネームスペース管理活動の間接的な結果として、スカルクを開始します。この活動には次のものが含まれます。
  - レプリカの追加または削除
  - ディレクトリーの作成または削除
  - レプリカ・タイプの再指定
  - 親セル内の子セル名の追加または削除

  これらの活動はすべてネームスペース構造の変更をもたらすので、即時スカルクを使用すれば、新しい構造をネームスペース全体に可能な限りすばやく反映させることができます。
- CDS サーバーは、*バックグラウンド・スカルク時間* と呼ばれるルーチン間隔で自動的にスカルクを開始します。

  バックグラウンド・スカルク時間間隔は、ネームスペース管理活動やユーザー開始のスカルクなどの他の要因とは関係なく、ディレクトリーのスカルク相互間の最大経過時間を保証します。 CDS サーバーはそのクリアリングハウス内の各マスター・レプリカを定期的に検査して、最後にスカルクが正常に実行されて以後に変更が行われたディレクトリーがあれば、そのスカルクを開始します。

タイム・スタンプがデータの整合性の保持にどのように役立つか

CDS は、データの整合性と正確さの保証を援助するために、いくつかのタイム・スタンプを使用します。各エントリーごとに、以下の 2 つのタイム・スタンプが存在します。

- 生成タイム・スタンプ (CTS)
- 更新タイム・スタンプ (UTS)

CDS は、セル・ネームスペース内のすべて (クリアリングハウス、ディレクトリー、オブジェクト・エントリー、ソフト・リンク、および子ポインター) に CTS を割り当てます。 CTS は、クリアリングハウス、ディレクトリー、またはディレクトリー内のエントリーが作成された日付、時刻、および場所を反映した、固有の値です。 CTS は 2 つの部分 (時刻部分とその名前が作成されたノードのシステム識別子) で構成されています。この 2 つの部分は、異なるノードで作成された複数のタイム・スタンプの間での固有性を保証します。

新しい名前または変更された名前がそれを作成したディレクトリーの各レプリカに伝搬される時、すべての CDS サーバーは、新しい名前を受け入れる前に CTS の妥当性を検査します。

UTS は、クリアリングハウス、ディレクトリー、オブジェクト・エントリー、ソフト・リンク、または子ポインターの属性に加えられた、最新の変更を反映していま

す。 CDS サーバーは、ディレクトリーの既存のエントリーに対する更新を受信すると、その更新を受け入れる前に、UTS の妥当性を検査します。

ディレクトリーとレプリカには他にもいくつかのタイム・スタンプがあり、 CDS はディレクトリーをスカルクするか、それともディレクトリーに変更を加えるかを判断するときに、これらを使用します。 (CDS によって使用される他のタイム・スタンプ属性については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

ダウンしたサーバーに対する修正内容

DCE for AIX V3.1 より前のリリースでは、ダウンしたクリアリングハウスと通信しようとすると長時間のタイムアウト遅延が生じました。 DCE for AIX および Solaris V3.1 では、このような遅延を防止するため、クライアントのコードにいくつかの変更を加えました。これらの変更箇所は **/etc/dce/cds.conf** ファイルで追加を行うことにより、指定変更または構成を行うことが可能です。

以下に示すのは、拡張内容の要約と、これらの変更箇所の指定変更または構成を行うための **cds.conf** ファイルの修正内容です。

- 周期的に実行することによってキャッシュに書き込まれたクリアリングハウスが使用可能であるかどうかを調べる、新しいスレッドが作成されます。すべてのクリアリングハウスが *ok* とマークされていることを検知すると、このスレッドは指定された時間だけスリープします。しかし、*not ok* とマークされたクリアリングハウスが見つかると、このスレッドは **rpc_mgmt_is_server_listening( )** 呼び出しによって該当するクリアリングハウスにコンタクトしようとします。サーバーが応答すると、その状況は *ok* に変更されます。

  次の検査までの遅延インターバルは、以下に示す書式の行を **/etc/dce/cds.conf** ファイルに追加することによって構成できます。

  ```
  cds.*.ch_wait_interval: value
  ```

  上記の例において、*value* は待機する秒数です。デフォルトは 120 秒になります。
- 選択したクリアリングハウスに cdsclerk が実際にコンタクトする前に、 cdsclerk はそのクリアリングハウスへのハンドルのリストを作成し、接続のないハンドルをリストの最初の部分に入れます。その後、**rpc_mgmt_is_server_listening()** 呼び出しを実行します。これらのハンドルの通信タイムアウトは、長い時間待機することがないように、低い値に設定されています。通信が確立できないとそのクリアリングハウスは *not ok* とマークされ、まだ他にもある場合は別の適当なクリアリングハウスを選んで検査を行います。

  さらに、この機能は **/etc/dce/cds.conf** ファイルに行を追加することによっても構成されます。フォーマットは次のとおりです。

  ```
  cds.*.process_addr_check: value
  ```

  上記の例において、 *value* は 1 (検査を実行する) または 0 (検査を実行しない) のいずれかを入れます。デフォルトでは、検査は実行されます。
- クライアントは、ダウンしているとマークされたサーバーにはコンタクトできないようになっています。 (他のベンダーの **rpc_mgmt** 呼び出しが正しく動作しないセルでは、これを無効にしなければならないことがあります。)

この機能を構成するには、以下の行を **/etc/dce/cds.conf** ファイルに追加します。

```
cds.*.skip_down_ch:  value
```

上記の例において、 *value* は 1 (*not ok* のクリアリングハウスをスキップする) または 0 (*not ok* のクリアリングハウスを検査する) となります。デフォルトは 1 です。

第15章 DCE ディレクトリー・サービスの管理

DCE 制御プログラム (**dcecp**) は、 CDS の管理に必要なほとんどのコマンドを提供します。この章では、 DCE 制御プログラムによって管理できる CDS エンティティーについて説明するとともに、これらのエンティティーの管理に有効なコマンドを要約します。

dcecp コマンドの詳細については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

DCE 制御プログラムの使用法

このガイドの 3ページの『第1章 DCE 制御プログラムの紹介』で、 **dcecp** とそのコマンド構文についてすでに紹介済みであるため、この章ではこの点に関しては繰り返しません。その代わりに、この章では、 CDS を管理するために **dcecp** が特に提供しているコマンドについて説明します。

CDS の管理するオブジェクト

DCE 制御プログラム・コマンドは、CDS エンティティーを表す、以下のオブジェクトに対して働きます。

directory
このオブジェクトは CDS ディレクトリーを表します。ディレクトリーは親ディレクトリーか子ディレクトリーの場合があり、あるいは親ディレクトリーか子ディレクトリーのマスターか読み取り専用レプリカの場合があります。 CDS ディレクトリーは、子ディレクトリーの他に、他の CDS リソースのソフト・リンクとオブジェクト・エントリーを含んでいる場合があります。

link このオブジェクトは CDS ディレクトリー内のソフト・リンクを表します。ソフト・リンクは、子ディレクトリー、オブジェクト・エントリー、または他のソフト・リンク (の代替名) に対するポインターです。

object このオブジェクトは、セル・ネームスペース内に現れる CDS リソースの名前である、オブジェクト・エントリーを表します。一部のオブジェクト・エントリーは、 CDS クライアントがアクセスできるリソース (たとえば、ディスク、マシン、あるいはアプリケーションなど) を指定します。その他のオブジェクト・エントリーは、 CDS が内部で使用するだけのリソース (たとえば、サーバーとクリアリングハウス) を指定します。

clearinghouse
このオブジェクトは CDS クリアリングハウスを表します。クリアリングハウスとは、サーバーが使用するために CDS サーバー・マシン上に置かれている、データベースのことです。

cdscache
このオブジェクトは CDS キャッシュを表します。 CDS キャッシュとは、CDS クラークが参照するためにローカル・システム上に設定した、サーバー、クリアリングハウス、およびその他の CDS リソースに関する情報の集合です。

cdsalias
このオブジェクトは、CDS が認識している、DCE セルの別名を表します。

注: cdsalias の作成と接続は、 IBM DCE V3.2 ではサポートされていません。 cdsalias の削除とカタログ作成がサポートされています。

cds このオブジェクトは CDS サーバーを表します。

cdsclient
このオブジェクトは CDS クライアントを表します。

CDS に対する DCE 制御プログラム・オペレーション

表8 は、 **dcecp** が CDS オブジェクト上で実行するオペレーションの一覧です。

表 8. CDS に対する DCE 制御プログラム・オペレーション

オペレーション	定義
add	セル・ネームスペース内の親ディレクトリーに子ディレクトリーを追加します。
catalog	DCE セルの別名またはクリアリングハウスのリストを表示します。
delete	セル・ネームスペース内のオブジェクトを削除します。オブジェクト・タイプにはディレクトリー、オブジェクト・エントリー、ソフト・リンク、クリアリングハウス、または CDS セル別名があります。
disable	クリアリングハウスの知識をローカル・マシン上で実行中のサーバーから削除するか、あるいは CDS サーバーまたは CDS クライアントを使用不能にします。
discard	CDS クライアントによって保持されているキャッシュ情報を完全に削除します。
dump	CDS キャッシュのコア内ダンプを表示します。
help	CDS オブジェクト・タイプに関して、それが実行するオペレーションまたはそれに対して実行できるオペレーションを記述したヘルプ・メッセージを表示します。オブジェクト・タイプはディレクトリー、オブジェクト・エントリー、ソフト・リンク、クリアリングハウス、または CDS キャッシュのいずれかです。
initiate	指定されたクリアリングハウス上で、特定のオペレーションを開始します。
list	ディレクトリーに含まれているすべての CDS オブジェクトの名前を表示します。
merge	ディレクトリーの内容を別のディレクトリーにコピーします。
modify	CDS オブジェクト・タイプに関する属性情報を修正します。オブジェクト・タイプはディレクトリー、オブジェクト・エントリー、またはソフト・リンクのいずれかです。
operations	CDS オブジェクト・タイプが実行できるかまたはそれに対して実行することができた、オペレーションを表示します。オブジェクト・タイプはディレクトリー、オブジェクト・エントリー、ソフト・リンク、またはクリアリングハウスのいずれかです。
ping	すべてのサーバーまたは選択されたサーバーが DCE セルの中で実行しているかどうかをチェックします。

表 8. CDS に対する DCE 制御プログラム・オペレーション (続き)

オペレーション	定義
remove	セル・ネームスペース内の親ディレクトリーから子ディレクトリーを削除します。
repair	指定されたクリアリングハウス上で、特定の診断オペレーションを開始します。
show	CDS オブジェクト・タイプに関する属性情報を表示します。オブジェクト・タイプはディレクトリー、オブジェクト・エントリー、ソフト・リンク、またはクリアリングハウスのいずれかです。
synchronize	子ディレクトリーまたは親ディレクトリーにそのレプリカと同期をとる (スカルクを実行する) ように命令します。

CDS オブジェクト属性

すべての CDS オブジェクトには属性があります。属性というのは、そのオブジェクトに関連したデータの断片または集合をいいます。属性は、オブジェクトのオペレーション動作を反映するかあるいはそれに影響を与えることができます。一部の属性は CDS によってしか作成あるいは変更することができません。その他の属性は使用する環境のニーズに応じて変更することができます。特定の CDS オブジェクトの属性の完全なリストについては、該当ページを参照してください。また、ほとんどのオブジェクトに対して **dcecp show** オペレーションを使用して、すべての属性の名前と値、またはオブジェクトの特定の属性を表示することができます。

CDS を保守するための dcecp の使用

特定の CDS 保守タスクに対して、**dcecp** を使用することができます。これを行うための **dcecp** コマンドが、表9 にリストされています。

表 9. CDS を制御する dcecp コマンド

コマンド	定義
cdsclient disable	CDS クラークの実行を停止します。
cds disable	CDS サーバーの実行を停止します。
directory modify	新しいマスター・レプリカを指定することにより、ディレクトリーのレプリカ・セットを作成し直します。
directory show	DNS または GDS 内でのセル・エントリーの作成に必要な情報を表示します。
cdsclient show	CDS クラークの属性を表示します。
cds show	CDS サーバーの属性を表示します。

第16章 CDS 名へのアクセスの制御

この章では、以下の CDS 認可について説明します。

- CDS に関する DCE 認可の概要
- CDS がサポートする DCE 認可のコンポーネント
- CDS がサポートする DCE 許可
- CDS のクラークおよびサーバー管理オペレーションに対するアクセスの制御
- 制御プログラム・コマンドと必要な許可
- CDS 名上での ACL の編集
- CDS サーバーがネームスペースに対するアクセス権を獲得する方法
- 新しいネームスペース内でのアクセス制御の設定

CDS に関する DCE 認可の概要

CDS 認可を使用すると、以下の CDS コンポーネントに対するユーザー・アクセスの制御が可能になります。

- クリアリングハウス、ディレクトリー、オブジェクト・エントリー、ソフト・リンク、および子ポインターを含めた、ネームスペースに格納されている名前
- 特権 CDS クラークおよびサーバー・コマンドの実行

ACL を作成することによって、ネームスペース内の名前に対するアクセスを制御します。 ACL には、 ACL が関連している名前に対してユーザー (プリンシパル) に与えた認可を指定する個々の ACL エントリーが含まれています。作成された ACL エントリーは、どのプリンシパルがその名前を使用できるか、およびその名前に対してどのような管理オペレーションを実行できるかを、全体として判別します。

CDS ACL 管理ソフトウェアは、 CDS のすべてのクラークとサーバーに組み込まれており、着信する CDS 要求についてアクセス検査を実行します。プリンシパルが CDS 名のオペレーションを要求すると、その名前が格納されているサーバーの ACL 管理ソフトウェアが、その名前と関連した ACL エントリーを調べます。その上で、ソフトウェアは、 ACL エントリーの中の要求元プリンシパルに対して与えられている認可に基づいて、オペレーションを認めるかまたは拒否します。同様に、プリンシパルが CDS のクラークまたはサーバーの特権オペレーションを要求すると、そのシステムの ACL 管理ソフトウェアが、このクラークまたはサーバーを表すプリンシパル名と関連した ACL エントリーを調べます。その上で、ソフトウェアは、 ACL エントリーの中の要求元プリンシパルに対して与えられている認可に基づいて、オペレーションを認めるかまたは拒否します。

DCE 制御プログラム (**dcecp**) には、CDS 名、クラーク、およびサーバーと関連した ACL の追加、修正、コピー、削除、および表示を行うコマンドが準備されています。これらのコマンドについての詳細は、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。この章の残りの部分では、特に CDS に適用した場合の DCE 許可について説明します。 CDS 名、ク

ラーク、またはサーバーに対する許可を作成または変更する前に、このガイドの 315ページの『第6部 DCE セキュリティー・サービス』を読んで、DCE 認可のメカニズムについての情報を得てください。

CDS によってサポートされる ACL タイプ

CDS は、以下の DCE ACL タイプをサポートします。

- オブジェクト ACL - オブジェクト ACL タイプを使用して、 CDS のクラークやサーバーに対してだけでなく、 CDS 名 (すなわち、オブジェクト・エントリー、ソフト・リンク、子ポインター、クリアリングハウス、およびディレクトリー) にも認可を与えることができます。 CDS ディレクトリーと関連している場合、オブジェクト ACL タイプに与えた認可は、そのディレクトリーの内容や子ディレクトリーではなく、そのディレクトリー自体にしか適用されません。
- 初期オブジェクト作成 ACL - 初期オブジェクト作成 ACL タイプは、 CDS ディレクトリー名にのみ適用されます。この ACL タイプは、ソフト・リンク、アプリケーション定義オブジェクト・エントリー、子ポインター、およびクリアリングハウス・オブジェクト・エントリーを含む、特に将来のディレクトリー内容に対する認可を与えるときに使用します。初期オブジェクト作成 ACL タイプを使用して与えられる認可は、そのディレクトリー自体ではなく、ディレクトリーの将来の内容にのみ適用されます。この許可は、ACL エントリーを作成した後でディレクトリー内に作成される名前のみによって継承され、すでにそのディレクトリー内に存在している名前にまで伝搬されることはありません。

  初期オブジェクト作成 ACL を編集するには、 **dcecp acl modify** コマンドの **-io** オプションを使用します。
- 初期コンテナー作成 ACL - 初期コンテナー作成 ACL タイプは、 CDS ディレクトリー名にのみ適用されます。この ACL タイプは、ディレクトリーに対する認可を与えるために使用されます。この認可は、後でそのディレクトリーの下に作成する可能性のあるすべての子ディレクトリーに自動的に伝搬されます (デフォルト)。初期コンテナー作成 ACL タイプを使用して与えられる認可は、 ACL エントリーを作成した後で 作成される子ディレクトリーのみによって継承され、すでに存在している子ディレクトリーには伝搬されません。

  初期コンテナー作成 ACL を編集するには、 **dcecp acl modify** コマンドの **-ic** オプションを使用します。

CDS ディレクトリーとその内容に許可を伝搬する方法

ディレクトリーの 3 つの ACL タイプ (オブジェクト ACL、初期オブジェクト作成 ACL、および初期コンテナー作成 ACL) をすべて作成することによって、ディレクトリー自体に限らず、そのディレクトリーの将来の内容と後で作成される可能性のある子ディレクトリー (およびその内容) に対するアクセスも認可することができます。

たとえば、**/.:/sales** という名前の新しいディレクトリーを作成したばかりであるとします。ユーザー **Smith** に **/.:/sales** ディレクトリーに対する読み取り認可を与える、オブジェクト ACL タイプの ACL エントリーを 1 つ作成すると、 **Smith** は、以下を行うことができます。

- **/.:/sales** ディレクトリーに関連する属性の読み取り

- **/.:/sales** ディレクトリーに格納された名前の表示

ユーザー **Smith** に **/.:/sales** ディレクトリーに対する読み取り認可を与える、初期オブジェクト作成 ACL タイプの 2 番目の ACL エントリーを作成すると、 **Smith** は、以下のことができます。

- **/.:/sales** ディレクトリーに関連する属性の読み取り
- **/.:/sales** ディレクトリーに格納された名前の表示
- 後で **/.:/sales** ディレクトリー内に作成される可能性のあるすべての名前と関連した属性の読み取り (ただし、作成後に明示的な ACL の変更によって禁止された場合を除く)

ユーザー **Smith** に、これも **/.:/sales** ディレクトリーに対する読み取り認可を与える、初期コンテナー作成 ACL タイプの 3 番目の ACL エントリーを作成すると、**Smith** は、以下を行うことができます。

- **/.:/sales** ディレクトリーと関連した属性の読み取り
- **/.:/sales** ディレクトリーに格納された名前の表示
- 後で **/.:/sales** ディレクトリー内に作成する可能性のあるすべての名前と関連した、属性の読み取り
- **/.:/sales** ディレクトリーの下に後で作成される可能性のある、すべての子ディレクトリーに対する上記の 3 つのオペレーションすべての実行

(デフォルト ACL の詳細については、このガイドの 315ページの『第6部 DCE セキュリティー・サービス』を参照してください。)

プリンシパルに使用される ACL エントリー・タイプ

ACL エントリーが作成されたプリンシパルのカテゴリーを指定するには、 ACL エントリー・タイプを使用します。これらの ACL エントリー・タイプについて、 表 10 で説明します。

表 10. CDS プリンシパルに使用される ACL エントリー・タイプ

エントリー・タイプ	目的
user	ローカル・セル内で証明書が認証された、個々のプリンシパルの ACL エントリーを指定します。
group	ローカル・セル内でメンバーが認証されている、許可グループの ACL エントリーを指定します。
other_obj	タイプ **user** の ACL エントリーによって指定された個々のユーザーでもなく、タイプ **group** の ACL エントリーによって指定されたグループのメンバーでもないプリンシパルで、ローカル・セル内で認証されているプリンシパルの ACL エントリーを指定します。
foreign_user	外部セル内の認証されているプリンシパルの ACL エントリーを指定します。
foreign_group	メンバーが外部セルで認証されている許可グループの ACL エントリーを指定します。

表 10. CDS プリンシパルに使用される ACL エントリー・タイプ (続き)

エントリー・タイプ	目的
foreign_other	タイプ **foreign_user** の ACL エントリーによって指定された個々のユーザーでもなく、タイプ **foreign_group** の ACL エントリーによって指定されたグループのメンバーでもないプリンシパルで、外部セルで認証されているプリンシパルの ACL エントリーを指定します。
any_other	上記の ACL エントリー・タイプではカバーされない認証プリンシパルの ACL エントリーを指定します。
mask_obj	プリンシパル (証明書は認証済みまたは未認証) の許可の代わりとなるマスクを含む、 ACL エントリーを指定します。
unauthenticated	認証手順にパスできないプリンシパルの ACL エントリーを指定します。
user_delegate	ローカル・セル内で認証されたプリンシパルの代理をする、仲介者の ACL エントリーを指定します。
group_delegate	ローカル・セル内の許可グループのメンバーである認証プリンシパルの代理をする、仲介者の ACL エントリーを指定します。
other_delegate	**user_delegate** タイプの ACL エントリーによって指定された個々のユーザーでもなく、**group_delegate** タイプの ACL エントリーによって指定されたグループのメンバーでもない、ローカル・セル内の認証プリンシパルの代理をする、仲介者の ACL エントリーを指定します。
foreign_user_delegate	外部セル内の認証プリンシパルの代理をする、仲介者の ACL エントリーを指定します。
foreign_group_delegate	外部セル内の許可グループのメンバーの代理をする、仲介者の ACL エントリーを指定します。
foreign_other_delegate	**foreign_user_delegate** タイプの ACL エントリーによって指定された個々のユーザーでもなく、**foreign_group_delegate** タイプの ACL エントリーによって指定されたグループのメンバーでもない、外部セル内の認証プリンシパルの代理をする、仲介者の ACL エントリーを指定します。
any_other_delegate	認証プリンシパルまたはグループの仲介者に対する他のどのタイプの ACL エントリーによっても指定されていない、ローカル・セルまたは外部セル内の認証プリンシパルの代理をする、仲介者の ACL エントリーを指定します。

CDS によってサポートされる DCE 許可

CDS は、読み取り (**r**)、書き込み (**w**)、挿入 (**i**)、削除 (**d**)、テスト (**t**)、制御 (**c**)、および管理 (**a**) の DCE 許可をサポートします。それぞれの許可は、関連付けられている CDS 名に応じて意味が多少異なります。一般に、許可は次のように定義されます。

- 読み取り許可 - プリンシパルに対して、名前の検索とそれに関連した属性値の表示を認めます。
- 書き込み許可 - プリンシパルに対して、 (その ACL を除く) 名前に関連した修正可能属性の変更を認めます。
- 挿入許可 - プリンシパルに対して、 (ディレクトリー・エントリーのみに使用するために) ディレクトリー内での新しい名前の作成を認めます。
- 削除許可 - プリンシパルに対して、ネームスペースからの名前の削除を認めます。
- テスト許可 - プリンシパルがどの値も見ることができなくても (つまり、名前の読み取り許可がなくても)、名前の属性に特定の値があるかどうかをテストすることを認めます。

  テスト許可は、より有効な CDS 属性値の検証方法をアプリケーション・プログラムに提供します。アプリケーションは、値の集合全体を読み取らなくても、特定の値が存在するかどうかをテストすることができます。
- 制御許可 - プリンシパルに対して、名前に関連した ACL エントリーの変更を認めます。 (CDS エントリーの ACL を変更するには読み取り許可も必要であり、それがなければ **dcecp** と **acl_edit** はエントリーにバインドできなくなります。) 制御認可は CDS エントリーの作成者に対して自動的に与えられます。
- 管理許可 - プリンシパルに対して、ディレクトリーの複製を制御する CDS コマンドの発行を認めます。管理許可は、ディレクトリー・エントリーでのみ使用されます。

プリンシパルは、名前について管理オペレーションを行うためには、その名前に対する何らかの許可を持っていなければなりません。許可がないと、プリンシパルが管理オペレーションを行おうとしたとき、 CDS は、名前を認識しないため、名前が存在しないというエラーを戻します。プリンシパルにある種の許可があってもそれがオペレーションの実行に必要な許可でない場合は、 CDS は、プリンシパルがそのオペレーションを行うのに十分な権限を持っていないことを説明するエラーを戻します。

名前の作成者には、作成した名前のタイプに適したすべての認可が自動的に与えられます。たとえば、オブジェクト・エントリーを作成するプリンシパルには、そのオブジェクト・エントリーに対する読み取り、書き込み、削除、テスト、および制御の認可が与えられます。ディレクトリーを作成するプリンシパルには、そのディレクトリーに対する読み取り、書き込み、挿入、削除、テスト、制御、および管理の認可が与えられます。

注: 他の大部分のファイル・システムで行われているセキュリティー・メカニズムとは異なり、 CDS では、名前に対してオペレーションを実行するために、プリンシパルはその名前のパス名 (フルネーム) のすべての中間要素に対するアクセス権を持っていなくてもかまいません。たとえば、 **/.:/sales** ディレクトリーに格納されているオブジェクト・エントリー **object1** を考えてみます。 CDS では、 **/.:/sales** ディレクトリーまたはセル・ルート・ディレクトリー (**/.:**) のいずれかに対するアクセス権をプリンシパルに与えなくても、オブジェクト・エントリー **/.:/sales/object1** に対するアクセス権をプリンシパルに与えることができます。

CDS のクラークおよびサーバー管理オペレーションに対するアクセスの制御

CDS 認可により、 CDS のクラークおよびサーバーに対するローカル管理オペレーションを伴う CDS コマンドの使用を制御することが可能となります。各クラークおよびサーバーのプリンシパル名は、セキュリティー・ネームスペースに格納されます。各クラークとサーバーのバインディング情報を含んでいるオブジェクト・エントリーは、 **/.:/hosts** サブディレクトリーの中の CDS ネームスペースに格納されます。サーバーは、**/.:/hosts/***hostname***/cds-server** として表されます。クラークは、**/.:/hosts/***hostname***/cds-clerk** として表されます。 (スリム・クライアントは、CDS ネームスペース内に表示されません。)

各クラークとサーバーは、これらのオペレーションの実行を許されたプリンシパルを指定した ACL エントリーを含んだ、個別の ACL を維持します。ネームスペース内の名前と関連している ACL と異なり、クラークやサーバーと関連している ACL は、これらのコマンドの使用をローカル制御するためだけに存在します。

新しいクラークまたはサーバーが初期化されるとき、常に ACL がクラークまたはサーバー・システム上に作成されます。初期 ACL エントリーも作成され、マシン・プリンシパルおよびネームスペース認可グループ (**subsys/dce/cds-admin**) に、そのシステム上のクラークまたはサーバー・プロセスに対する読み取り、書き込み、および制御の認可が与えられます。他のすべてのプリンシパル (認証済みも未認証も含む) には、読み取り認可が与えられます。この ACL エントリーを作成すると、作成直後に、システムにマシン・プリンシパルとしてログインしたすべてのユーザーに、特権クラークまたはサーバー CDS コマンドを実行する許可が与えられます。

注: この目的でマシン・プリンシパルを使用するのは便利さのためであり、この場合、アカウント自体 (ユーザー名とパスワード) はすでに適度に保護されているものと想定しています。ネームスペース管理者は、この方式を変更し、他の個々のプリンシパルまたは認可グループの代わりに、特定のクラークおよびサーバーに認可を与えることもできます。

CDS のクラークまたはサーバーに関連する ACL を編集するためには、 **-change** オプションを指定した **dcecp acl modify** コマンドを使用します。たとえば、ノード **orion** 上の CDS クラークと関連した ACL 内のユーザー **michaels** の許可を変更するには、以下のコマンドを入力します。

```
dcecp> acl modify /.:/hosts/orion/cds-clerk -change {user michaels rw}
dcecp>
```

クラークとサーバーは、ネームスペース内のエントリーによっても表されることに注意してください。 CDS クラークまたはサーバーのネームスペース・エントリーと関連した ACL を編集するには、 **acl modify** コマンド行に、**-change** オプションだけでなく、 **-entry** オプションも指定しなければなりません。 DCE リソースの CDS エントリー上の ACL を修正する方法の詳細については、 203ページの『CDS 名上での ACL の編集』を参照してください。

制御プログラム・コマンドと必要な許可

表11 は、CDS オブジェクトに作用する **dcecp** コマンドのすべてと、そのコマンドを実行するためにプリンシパルが持っていなければならない許可の一覧です。

表 11. DCE 制御プログラム・コマンドと必要な許可

コマンド	必要な許可
cds disable	サーバーのネームスペース・エントリー上での削除、書き込み、および作成の許可。
cds show	サーバーのネームスペース・エントリー上での読み取り許可。
cdsalias catalog	別名をリストしたいセルのルート・ディレクトリーに対する読み取り許可。
cdsalias delete	セルのルート・ディレクトリー上での auth_info 許可。
cdscache create	ローカル CDS キャッシュ内にサーバー・エントリーを作成するクラークに対する書き込み許可。
cdscache delete	ローカル CDS キャッシュ内のサーバー・エントリーから削除されるクラークに対する書き込み許可。
cdscache discard	CDS キャッシュが置かれているクラーク・システム上でのスーパーユーザー (ルート) 特権。 CDS の許可は必要ありません。
cdscache dump	CDS キャッシュが置かれているクラーク・システム上でのスーパーユーザー (ルート) 特権。 CDS の許可は必要ありません。
cdscache show	CDS キャッシュから、サーバー (-server オプション) 情報またはクリアリングハウス (-clearinghouse オプション) 情報のいずれかを検索するよう指示されたクラークに対する読み取り許可。
cdsclient disable	クラークのネームスペース・エントリー上での削除、書き込み、および作成の許可。
cdsclient show	ネームスペース・エントリー上での読み取り許可。
clearinghouse catalog	特別の特権は必要ありません。
clearinghouse create	クリアリングハウスを作成したいサーバーに対する書き込み許可、およびセル・ルート・ディレクトリーに対する管理許可。また、サーバー・プリンシパルには、セル・ルート・ディレクトリーに対する読み取り、書き込み、および管理の許可も必要です。
clearinghouse delete	削除されるクリアリングハウスに対する書き込み許可と削除許可、およびクリアリングハウス内にレプリカを保管する、すべてのディレクトリーに対する管理許可。また、サーバー・プリンシパルには、関連するクリアリングハウス・オブジェクト・エントリーに対する削除許可、およびクリアリングハウス内にレプリカを格納する、すべてのディレクトリーに対する管理許可も必要です。
clearinghouse disable	クリアリングハウスが常駐する CDS サーバーに対する書き込み許可。

表 11. DCE 制御プログラム・コマンドと必要な許可 (続き)

コマンド	必要な許可
clearinghouse initiate	クリアリングハウス・サーバー上での書き込み許可、およびセル・ルート・ディレクトリー上での管理許可。サーバー・プリンシパルには、セル・ルート・ディレクトリー上での読み取り、書き込み、および管理の許可が必要です。
clearinghouse repair	クリアリングハウス・サーバーに対する書き込み許可、およびセル・ルート・ディレクトリーに対する管理許可。サーバー・プリンシパルには、セル・ルート・ディレクトリーに対する読み取り、書き込み、および管理の許可が必要です。
clearinghouse show	属性をリストしたいクリアリングハウスに対する読み取り許可。
clearinghouse verify	クリアリングハウス・サーバーに対する書き込み許可、およびセル・ルート・ディレクトリーに対する管理許可。サーバー・プリンシパルには、セル・ルート・ディレクトリーに対する読み取り、書き込み、および管理の許可が必要です。
directory add	子ポインター (-member オプション) が置かれる親ディレクトリーに対する挿入許可。
directory create	親ディレクトリーに対する挿入許可と読み取り許可、および新しいディレクトリーのマスター・レプリカを格納する、クリアリングハウスに対する書き込み許可。また、サーバー・プリンシパルには新しいディレクトリーの親ディレクトリーに対する読み取りおよび挿入許可が必要です。
directory delete	ディレクトリーに対する削除許可、およびディレクトリーのマスター・レプリカを格納する、クリアリングハウスに対する書き込み許可。サーバー・プリンシパル (**hosts/***hostname***/cds-server**) には、親ディレクトリーに対する管理許可、および削除したいディレクトリーをポイントする、子ポインターに対する削除許可が必要です。
directory list	内容をリストしたいディレクトリーに対する読み取り許可。
directory merge	マージする元のディレクトリーと宛先のディレクトリーに対する読み取り許可、および宛先のディレクトリーに対する挿入許可。
directory modify	属性または属性値を追加 (-add オプション)、変更 (-change オプション)、または削除 (-remove オプション) したい、ディレクトリーに対する書き込み許可。
directory remove	子ポインター (-member オプション) に対する削除許可、または親ディレクトリーに対する管理許可。

表 11. DCE 制御プログラム・コマンドと必要な許可 (続き)

コマンド	必要な許可
directory show	属性をリストしたいディレクトリーに対する読み取り許可。ディレクトリーのレプリカの場合 (-replica オプション) - そのレプリカがメンバーである、ディレクトリーに対する読み取り許可。子ディレクトリーの場合 (-member オプション) - 子ディレクトリーに対する読み取り許可。
directory synchronize	ディレクトリーに対する管理、書き込み、挿入、および削除の許可。また、サーバー・プリンシパルには、ディレクトリーに対する管理、読み取り、および書き込みの許可も必要です。
link create	リンクをその中に作成しようとする、ディレクトリーに対する挿入許可。
link delete	リンク・エントリーに対する削除許可、または削除するリンク・エントリーを格納しているディレクトリーに対する管理許可。
link modify	属性を変更しようとするリンクに対する書き込み許可。
link show	属性をリストしようとするリンクに対する読み取り許可。
object create	オブジェクト・エントリーを格納する親ディレクトリーに対する挿入許可。
object delete	オブジェクト・エントリーに対する削除許可、またはそのオブジェクト・エントリーを格納する親ディレクトリーに対する管理許可。
object modify	属性または属性値を追加 (-add オプション)、変更 (-change オプション)、または削除 (-remove オプション) したい、オブジェクト・エントリーに対する書き込み許可。
object show	属性をリストしたいオブジェクト・エントリーに対する読み取り許可。

CDS 名上での ACL の編集

子ディレクトリー、クリアリングハウス、ソフト・リンク、またはその他の一部の CDS オブジェクトのための CDS ネームスペース内のエントリーと関連する ACL を編集するには、どの **dcecp acl** コマンドにも **-entry** オプションを指定します。**-entry** オプションは、あいまいなパス名の場合に特に便利です。パス名を DCE ディレクトリー・サービスのリーフ・オブジェクトおよび ACL をサポートする、別の DCE コンポーネント内のオブジェクトに分解することができる場合があります。このような場合、 **-entry** オプションを使用して、CDS 内のリーフ・オブジェクトを編集しなければなりません。実際のクリアリングハウスまたはディレクトリーと関連した ACL の編集には、このオプションを指定する必要はありません。

たとえば、**/.:/Paris1_CH** という名前のクリアリングハウスの CDS エントリーと関連したオブジェクト ACL 内の許可を編集する場合は、以下のコマンドを入力します。

```
dcecp> acl modify /.:/Paris1_CH -entry -change {unauthenticated -}
dcecp>
```

/.:/Paris1_CH というクリアリングハウス自体と関連したオブジェクト ACL 内の許可を編集する場合は、以下のコマンドを入力します。

```
dcecp> acl modify  /.:/Paris1_CH  -change {unauthenticated -}
dcecp>
```

もう 1 つは、ソフト・リンク **/.../eng_printer** の例です。このソフト・リンクのターゲットは、 **/.../boston.com/print_server** です。 CDS ネームスペース内のソフト・リンクのリーフ・エントリーを編集するには、以下のコマンドを入力します。

```
dcecp> acl modify /.../eng_printer -change -entry ¥
> {group subsys/dce/cds-admin rwdtc}
dcecp>
```

CDS サーバーがネームスペースに対するアクセス権を獲得する方法

CDS サーバーには、次の CDS コマンドを正常に実行するために、セル・ルート・ディレクトリーおよび下位ディレクトリーに対する許可が必要です。

- **clearinghouse create**
- **directory create** (ディレクトリーおよびレプリカに対する)
- **directory delete** (ディレクトリーおよびレプリカに対する)
- **directory synchronize**

すべての CDS サーバーに必要な認可を与えるプロセスを自動化するために、 CDS セル構成プロセスは、固定名 **subsys/dce/cds-servers** の下に CDS サーバーに対する認可グループを作成します。セル内の初期サーバーのプリンシパル名が、構成プロセスの一部として、このグループに追加されます。グループが作成されると直ちに、構成プロセスは、新しいネームスペースのセル・ルート・ディレクトリーに対する全認可 (**r**、**w**、 **i**、**d**、**t**、**c**、**a**) をグループのために与えます。オブジェクト ACL タイプと初期コンテナー作成 ACL タイプの ACL エントリーは、各 ACL エントリー内のプリンシパルとして **subsys/dce/cds-servers** を指定することによって作成されます。これによって、グループが将来作成されるすべてのディレクトリーとその内容に対する完全アクセス権を持つことが保証されます。

その後は、新しいサーバーがセル内に構成されるときに、サーバー構成プロセスが自動的に新しいサーバーのプリンシパル名をグループに追加します。このプロセスによって、セル内のすべての CDS サーバーは、ネームスペース内のすべてのディレクトリーに対する適切な許可を受け取ります。

新しいネームスペース内でのアクセス制御の設定

最初の CDS サーバーを構成したら直ちに、新しいディレクトリーを作成あるいは転送する*前*に、整合性のあるアクセス制御方針を計画し、その方針を実現する準備をしておかなければなりません。実行できるタスクとしては、以下のものがあります。

- ネームスペース許可グループへのメンバーの追加

- 追加の許可グループの作成
- 未認証プリンシパルに対する最大許可の設定

ネームスペース許可グループへのメンバーの追加

ネームスペースの管理とトラブルシューティングを容易にするために、セル構成プロセスは、固定名 **subsys/dce/cds-admin** の下にネームスペース許可グループを作成します。その後、構成プロセスは、セル・ルート・ディレクトリーに対するグループ最大アクセス権をそのグループに与えます。このアクセス権は、ネームスペースの展開に伴ってネームスペース全体に伝搬されます。

作成直後は、許可グループには、初期ネームスペース管理者がセル構成プロセスの間に指定した名前しか入っていません。 **dcecp group add** コマンドを使用して、ネームスペースの管理とトラブルシューティングを行う、オーガニゼーション内の他の人のプリンシパル名を追加することができます。このグループはネームスペース全体に対する完全アクセス権を持っているので、そのメンバーは、必要な時にはいつでも、限られた許可しか持たないネームスペース・ユーザーに代わって、問題を解決するために介入することができます。ユーザーのプリンシパル名をグループから削除すると、そのプリンシパルによって記述されたユーザーはそのグループに割り当てられていたアクセス権を失います。

(グループ・メンバーの追加および削除の方法の詳細については、このガイドの315ページの『第6部 DCE セキュリティー・サービス』を参照してください。)

追加の許可グループの作成

許可グループは、ネームスペースへのアクセスを制御するための、便利で柔軟性のある方法を提供することができます。ユーザーをオーガニゼーション、作業タイプ、セキュリティーの状況などに従って結合し、その後、特定のディレクトリーまたはネームスペース内のその他の名前に対する特定の認可セットを、各グループに対して与えることができます。

ローカルに権限を委譲するために、ネームスペース内に作成を予定している、それぞれの機能ディレクトリーについて許可グループを作成することができます。たとえば、**subsys/dce/sales-admin** という名前の許可グループを作成して、そのメンバーとして、 **/.:/sales** ディレクトリーを管理する責任を持つ人を含めることができます。それぞれのローカル許可グループは、責任を持つディレクトリーの内容に対して完全なアクセス権を持つことができます。

未認証プリンシパルに対する最大許可の設定

すべての未認証プリンシパルに対するネームスペース全体の最大の許可セットを適用したい場合、最初の CDS サーバーを構成した*直後* で、しかもそのセル・ルートの下にいずれかのディレクトリーを作成あるいは転送する*前*に、それを行わなければなりません。オブジェクト ACL タイプと初期コンテナー作成 ACL タイプを使用して未認証 ACL エントリーおよびそのセル・ルートの **any_other** エントリーを作成することにより、未認証エントリーの展開に伴うネームスペース全体への未認証エントリーの自動伝搬を利用することができます。

第17章 クラーク、サーバー、およびクリアリングハウスの管理

CDS のクラーク、サーバー、およびクリアリングハウスは、最初は CDS のクラークとサーバーの構成の一部として作成され、開始されます。その後、クラークとサーバーのプロセスは、手動で実行されるか、あるいは動作しているシステム上の起動スクリプトによって実行される、一連のコマンドを使用して作成され、開始されます。このような CDS エンティティーは大部分が自己規制的であり、ルーチンのモニタリングを除き、管理のための介入は少ししか必要としません。

この章では、CDS クラーク、サーバー、およびクリアリングハウスのモニター方法と、その他の管理タスク (ネームスペース情報のバックアップなど) の実行方法について説明します。

クラーク、サーバー、およびクリアリングハウスのカウンターのモニタリング

各クラーク、サーバー、およびクリアリングハウスは、最後の始動以後に実行したか、またはそれらに対して実行された、読み取り、書き込み、およびその他のオペレーションを追跡するために、カウンター と呼ばれる、1 組の属性を維持しています。これらのカウンターをモニターすることにより、ネットワーク上で生成される CDS トラフィックのタイプや量を判別することができます。

クラーク、サーバー、およびクリアリングハウスのカウンターについては、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* に詳しく説明されています。

クラーク・カウンターの表示

dcecp cdsclient show コマンドを使用して、クラークの現行のカウンター値を表示します。たとえば、あるクラークに関連するすべての属性の現行値を表示するには、以下のコマンドを入力します。

```
dcecp> cdsclient show /.:/hosts/<hostname>/cds-clerk
```

サーバー・カウンターの表示

dcecp cds show コマンドを使用して、サーバーの現行のカウンター値を表示します。たとえば、あるサーバーに関連するすべての属性の現行値を表示するには、以下のコマンドを入力します。

```
dcecp> cds show /.:
```

クリアリングハウス・カウンターの表示

-counters オプションを指定した **dcecp clearinghouse show** コマンドを使用して、指定されたクリアリングハウスに対する現行のカウンター値を表示します。たとえば、以下のコマンドは、リモートのクリアリングハウス **/.:/Paris1_CH** に関連するすべての属性の現行値を表示します。

```
dcecp> clearinghouse show /.:/Paris1_CH  -counters
```

クラークと特定クリアリングハウスとの通信の設定

優先指定は、各クリアリングハウスにランクを割り当てることによって行われます。ランクは、16 ビットの符号なし整数 (範囲 0 ～ 65535) です。低い数字のほうが高い数字より優先されます (65535 というランクは、*このクリアリングハウスを決して使用しない* ことを意味します)。これらのランクは、**$DCELOCAL/etc/cds_serv_pref** というテキスト設定ファイル内に指定されます。ファイルのフォーマットは、ファイルの各行に 1 つのクリアリングハウス名と 1 つのランクが設定される形式です。ブランク行および注釈 (行の末尾に # が付いている) は、無視されます。ランクには 0 ～ 65535 (0x0000 ～ 0xFFFF) を指定することができ、 10 進数、8 進数 (先行 **0** を持つ)、または 16 進数 (先行 **0x** を持つ) で指定できます。クリアリングハウス名は、以下のどのフォーマットでも可能です。

```
/.../cellname/foo_ch
/foo_ch
foo_ch
/.:/foo_ch
```

クリアリングハウスのセル名が指定されない場合、ローカル・セルが指定されたものと想定されます。

```
Example file:
/.:/foo_ch  50  # most preferred clearinghouse
/.:/bar_ch  100
/.../mycellname/baz_ch 100
```

設定の変更ファイルでクリアリングハウスの指定がない場合、クリアリングハウスのためのランクが計算されます。 (したがって、デフォルトのランクがオーバーライドされることになっているクリアリングハウスのランクのみを指定する必要があります)。デフォルトのランクは、IP アドレスに基づいて計算されます。

- ローカル・ホスト・アドレスと一致するアドレスを持つクリアリングハウスは、5000 というデフォルトのランクを受け取ります。
- ローカル・ホストと同じ IP サブネット上のクリアリングハウスは、 20000 というデフォルトのランクを受け取ります。
- ローカル・ホストと同じ IP ネットワーク上のクリアリングハウスは、 3000 というデフォルトのランクを受け取ります。
- その他すべてのクリアリングハウスは、40000 というデフォルトのランクを受け取ります。

クリアリングハウスの設定の変更ファイルは、 **cdsadv** の起動時に読み取られ、値はキャッシュされます。ランク値を変更する場合、cds クライアントを停止し、キャッシュを削除してから、 **cds** クライアントを再起動しなければなりません。

以下のコマンドには、ランク属性が含まれます。

```
dcecp -c cdscache show -clearinghouse /.:/foo_ch
cdscp show cached clearinghouse /.:/foo_ch
```

クラークと特定クリアリングハウスとの通信のモニタリング

すべての CDS クラークは、別セットのクリアリングハウス・カウンターを維持しながら、通信相手の各クリアリングハウスに指示した読み取り、書き込み、およびその他のオペレーションを追跡します。これらのレコードが集まって、特定クラークのキャッシュ・クリアリングハウス・エンティティーを表します。

クラークのキャッシュに書き込まれたクリアリングハウス・カウンターをモニターしていれば、このクラークが使用する各クリアリングハウスにトランザクションがどのように分散されるかを調べて、クラークの要求が一番多く送られる先を知ることができます。これを行うためには、 **-clearinghouse** オプションを指定した **dcecp cdscache show** コマンドを使用します。たとえば、**/.:/NY1_CH** クリアリングハウスのローカル・クラークによって維持される、キャッシュ・クリアリングハウス・カウンターを表示するには、以下のコマンドを入力します。

```
dcecp> cdscache show  /.:/NY1_CH -clearinghouse
```

クリアリングハウスの内容の表示

dcecp clearinghouse show コマンドを使用して、特定のクリアリングハウスに格納されているすべてのディレクトリーのディレクトリー名を表示します。たとえば、クリアリングハウス **/.:/Chicago2_CH** に格納されているディレクトリーの名前を表示するには、以下のコマンドを入力します。

```
dcecp> clearinghouse show /.:/Chicago2_CH
```

(クリアリングハウス情報を表示するための別の例については、 225ページの『第19章 ネームスペースの構造と内容の表示』を参照してください。)

強制的なクリアリングハウスのディスクへのチェックポイント

通常のオペレーションでは、サーバーは定期的にメモリーからディスクへクリアリングハウスのチェックポイントをとります。しかし、サーバーに対する書き込み許可を持ち、 **checkpoint** オプションを指定した **dcecp clearinghouse initiate** コマンドを入力すれば、この作業を直ちに行うことができます。たとえば、メモリーからディスクへクリアリングハウス **/.:/Boston3_CH** のチェックポイントをとるには、以下のコマンドを入力します。

```
dcecp> clearinghouse initiate /.:/Boston3_CH -checkpoint
```

クラークとサーバーの使用不能化

時には、クラークまたはサーバーの活動中のプロセスの中断を必要とする、診断またはトラブルシューティング作業を行うために、特定のシステムで実行中のクラークまたはサーバーを使用不能にしなければならない場合があります。通常、**dce_config** プロシージャーを使用して、 DCE デーモンの起動および停止を行うことができます。 **dcecp** コマンドの **cdsclient disable** および **cds disable** を使用して、 CDS のクラークおよびサーバーを使用不能にすることができます。

クラークの使用不能化

ローカル・ノード上のクラークを使用不能にするには、次のコマンドを入力します。

```
dcecp> cdsclient disable /.:/hosts/<hostname>/cds-clerk
```

サーバーの使用不能化

ローカル・ノード上のサーバーを使用不能にするには、次のコマンドを入力します。

```
dcecp> cds disable /.:/hosts/<hostname>/cds-server
```

クラークとサーバーの再起動

CDS のクラークとサーバーのプロセスは、ホスト・システムがリブートされたときに必ず実行する、起動スクリプトによって自動的に作成され、起動されます。ただし、クラークまたはサーバーがリブート時に自動的な起動に失敗した場合、あるいはホスト・システムでバックアップあるいは診断作業を行うために使用不能にした、クラークまたはクラークを再起動する場合は、これらのスクリプトを手動で実行しなければなりません。 DCE デーモンの起動には、 **start.dce** コマンドが使用されます。

クラークの再起動

クラークを再起動するには、以下のステップに従います。

1. スーパーユーザー (**root**) としてクラーク・システムにログインします。
2. 次のコマンドを入力して、**dced** プロセスがすでに実行中かどうかを調べます。

   ```
   # ps -e
   ```

3. **dced** プロセスが活動プロセスのリストにある場合は、ステップ 4 に進んでください。 **dced** プロセスが活動プロセスのリストにない場合は、以下のコマンドを入力して、プロセスを起動します。

   ```
   # dced
   ```

4. 次のコマンドを入力して **cdsadv** プロセスを開始します。

   ```
   # cdsadv
   ```

スリム・クライアント構成でのクラークの再起動

スリム・クライアント構成でクラークを再起動するには、以下のステップに従います。

1. スーパーユーザー (**root**) としてクラーク・システムにログインします。
2. 次のコマンドを入力して **cdsclerk** プロセスを開始します。

   ```
   # cdsclerk -n
   ```

サーバーの再起動

サーバーを再起動するには、以下のステップに従います。

1. サーバー・システムにスーパーユーザー (**root**) としてログインします。
2. 次のコマンドを入力して、**dced** プロセスがすでに実行中かどうかを調べます。

   ```
   # ps -e
   ```

3. **dced** プロセスが活動プロセスのリストにある場合は、ステップ 4 に進んでください。 **dced** プロセスが活動プロセスのリストにない場合は、以下のコマンドを入力して、プロセスを起動します。

   ```
   # dced
   ```

4. 以下のコマンドを入力して、**cdsadv** プロセスがすでに実行中かどうか調べます。

   ```
   # ps -e
   ```

5. **cdsadv** プロセスが活動プロセスのリストにある場合は、ステップ 6 に進んでください。 **cdsadv** プロセスが活動プロセスのリストにない場合は、以下のコマンドを入力して、プロセスを起動します。

   ```
   # cdsadv
   ```

6. 以下のコマンドを入力して、サーバーを再起動します。

   ```
   # cdsd
   ```

サーバー・プロセスが起動すると、システム上のすべてのクリアリングハウスが起動します。

サーバー・システムの上位移行時のクリアリングハウスの保存

CDS サーバー・システム上のオペレーティング・システム・ソフトウェアの上位移行を計画する場合、そのシステム上のクリアリングハウスを保存したい場合は、次の手順に従ってください。

1. クラークとサーバーを使用不能にしていることを確認します。
2. システムの上位移行を行う前に、以下の CDS ファイルをバックアップします。
   - **$DCELOCAL/etc/cds_attributes**
   - **$DCELOCAL/var/adm/directory/cds/cds_files**
   - **$DCELOCAL/var/adm/directory/cds/*_ch.checkpoint** *nnnnnnnn*
   - **$DCELOCAL/var/adm/directory/cds/*_ch.tlog** *nnnnnnnn*
   - **$DCELOCAL/var/adm/directory/cds/*_ch.version**
   - **$DCELOCAL/var/directory/cds/cds_cache.** *nnnnnnnn*
   - **$DCELOCAL/var/directory/cds/cds_cache.version**
   - **$DCELOCAL/var/directory/cds/cds_cache.wan**

   (すべての CDS ファイルの完全パス名については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド － 入門* と *OSF DCE Porting and Testing Guide* を参照してください。)
3. システムの上位移行を実行します。
4. ステップ 2 でバックアップしたファイルをすべてリストアします。
5. 210ページの『クラークとサーバーの再起動』で説明されたサーバーの再起動のための手順に従います。サーバー・プロセスは、起動すると、該当のリストア済みファイルを自動的に見つけて、そのシステム上のすべてのクリアリングハウスを起動します。

ネームスペース情報のバックアップ

ディレクトリーの更新とスカルクは非同期的に発生する可能性があり、またネームスペースは分散する性質があるので、 CDS データの保存に従来のバックアップ方法を常に使用できるとは限りません。

この章の残りの部分では、以下のバックアップ・メカニズムをどの時点で使用するかについて説明します。

- ディレクトリー複製
- オペレーティング・システムのバックアップ

ネームスペース情報をバックアップするための複製の使用

ディレクトリー複製は、常に、ネームスペース内の情報をバックアップするための最も信頼できる方法です。クリアリングハウスにあるディレクトリーの新しいレプリカを作成すると、情報が配布されるばかりでなく、最新かつリアルタイムの情報のバックアップが作成されます。あるクリアリングハウスのレプリカが使用不能になった場合、ユーザーは別のクリアリングハウスにあるディレクトリーの別のレプリカ内で、必要とする情報を検索することができます。作成するディレクトリーのレプリカが増えるほど、ユーザーが常にネームスペースのどこかでディレクトリーに含まれている情報を検出できる可能性が高くなります。

クリアリングハウス全体が破壊された場合は、新しいクリアリングハウスを作成し、そこに格納されていたディレクトリーの新しいレプリカを作成することによって、それをリストアすることができます。 (レプリカの作成方法の詳細については、215ページの『第18章 CDS ディレクトリーの管理』を参照してください。)

オペレーティング・システムのバックアップの使用

ネームスペースは可変の間隔で修正が同期化される分散データベースであるため、ある特定のサーバー・システムの従来のバックアップはすべて、常に、古くて、不完全な情報を含んでいます。名前を頻繁に作成、変更、あるいは削除する場合、古くなったバックアップを復元すると、最近作成された名前が消えたり、最近の修正が元に戻ったり、あるいは最近削除された名前がネームスペースに再び表示されたりすることがあります。従来のバックアップがクリアリングハウスの現状をどの程度反映しているかは、以下の条件に完全に依存します。

- 最近のバックアップが作成された時期
- 作成後に行われた変更の内容
- ディレクトリー *dcelocal***/var/directory/cds** の中のクリアリングハウス・ファイルがバックアップに含まれたかどうか

オペレーティング・システムのバックアップを使用することに決めた場合、必要なのは、クリアリングハウスにディレクトリーのマスター・レプリカを格納しているサーバー・システムのバックアップだけです。ネームスペースを完全にバックアップするためには、以下についてチェックしてください。

- これらのシステム上のサーバーが、 **dcecp cds disable** コマンドを使用して使用不能にされている。
- ルート・ディレクトリー *dcelocal***/var/directory/cds** の中のファイルが、バックアップに含まれている。

ネームスペースが十分小さくて 1 つのクリアリングハウス内で維持できる場合は、従来のオペレーティング・システム・バックアップを安心して使用し、クリアリングハウス・データを保管あるいはリストアすることができます。クリアリングハウスが 1 つだけ存在する場合は、各ディレクトリーのレプリカ (マスター・レプリカ) が 1 つだけ存在します。この場合は、複数のディレクトリー・レプリカの間に存在する可能性のある、矛盾の原因を明らかにする必要がなくなります。クリアリングハウス・データのバックアップを頻繁に行えば行うほど、リストアが必要になった場合に情報がより最新のものになります。

再マスター化のパフォーマンスの向上

転送元 (現行マスター) のクリアリングハウスのタイム・スタンプが、転送先の (マスターにする) クリアリングハウスのタイム・スタンプより遅れていると、ディレクトリーのマスター・コピーを新しいクリアリングハウスに転送する際に、その完了まで非常に時間がかかることになります。再マスター化の際のパフォーマンスを向上するには、転送元のクリアリングハウスのタイム・スタンプが、転送先のクリアリングハウスのタイム・スタンプより進んでいるようにしてください。

第18章 CDS ディレクトリーの管理

25 ノード以下の小規模で成長の遅いネットワーク内のネームスペースを管理する場合は、自分の作成した名前をすべてルート・ディレクトリーで維持することができ、追加のディレクトリーを作成する必要はありません。しかし、25 ノードを超えるネットワーク内のネームスペースを管理する場合は、ルートの下に少なくとも 1 つの追加のディレクトリーのレベルを作成することを考慮してください。

ここでは、セル・ネームスペース内のディレクトリー階層を作成する方法について説明し、以下のようなディレクトリー管理に関する作業について解説します。

- ディレクトリーのレプリカの作成と削除
- ディレクトリーのスカルク
- ディレクトリーのコンバージェンスの変更

ディレクトリーの作成

ディレクトリーを作成すると、オブジェクト・エントリーのグループを、使用の場所、頻度、および使用する人にしたがって、複製したり管理したりすることができます。関連するオブジェクト・エントリーを別のディレクトリーにグループ分けすることによっても、デフォルトの ACL エントリーの伝搬を利用できるようになるので、アクセスの制御が容易になります。

CDS セル構成がルートの下にディレクトリーの初期階層を作成するので、これによって DCE コンポーネントは、ネームスペース内でオブジェクト・エントリーを作成しカタログすることのできる、位置を決めることができます。セル構成によって作成されたディレクトリーの中に **subsys** ディレクトリーがあり、この下に独立ソフトウェア・ベンダー (ISV) は独自のディレクトリーを作成して、自身の分散アプリケーションで使用するオブジェクト・エントリーを格納することができます。

あるいは、ISV およびネームスペースのその他のユーザーは、ルートの下に独自に設計したディレクトリーの階層を作成して、自身の情報を格納することもできます。

(セル構成によって設定された初期階層についての詳細は、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド － 入門* を参照してください。)

ディレクトリーを作成するための許可

ディレクトリーを作成するには、以下の許可が必要です。

- 新規ディレクトリーの親に対する挿入許可。
- 新規ディレクトリーのマスター・レプリカを格納する、クリアリングハウスに対する書き込み許可。
- DCE 制御プログラム (**dcecp**) の **directory create** コマンドを入力するサーバー・システムのサーバー・プリンシパルには、新規ディレクトリーの親ディレクトリーに対する読み取り許可と挿入許可が必要です。

  サーバーがサーバー許可グループ **subsys/dce/cds-servers** に含まれている場合、これらの許可はすでに与えられているはずです。疑わしい場合は、親ディレ

クトリーで **dcecp acl show** コマンドを使用して、サーバー・プリンシパルが適切な許可を得ていることを確認してください。 (**acl show** コマンドの引き数について詳しくは、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

directory create コマンドの入力

directory create コマンドを使用して、指定された名前を持つ新規ディレクトリー (マスター・レプリカ) を作成します。このコマンドを使用すると、CDS は、デフォルトで、新規ディレクトリーのマスター・レプリカを、新規ディレクトリーの親ディレクトリーのマスター・レプリカと同じクリアリングハウスに格納します。

たとえば、**/.:/sales** という名前のディレクトリーを作成して、新規ディレクトリーのマスター・レプリカをルート・ディレクトリーの初期クリアリングハウスに格納するには、以下のコマンドを入力します。

```
dcecp> directory create /.:/sales
dcecp>
```

注: ディレクトリーの作成を成功させるには、新規ディレクトリーの親ディレクトリーのマスター・レプリカがコマンド入力時に使用可能でなければなりません。

directory create コマンドの **-clearinghouse** オプションを使用して、新規ディレクトリーのマスター・レプリカを親ディレクトリーのクリアリングハウスとは別のクリアリングハウスに格納することができます。たとえば、上記の例で作成された新規ディレクトリーを別のクリアリングハウス (**/.:/Chicago1_CH**) に入れるには、以下のコマンドを入力します。

```
dcecp> directory create /.:/sales -clearinghouse /.:/Chicago1_CH
dcecp>
```

(**directory create** コマンドの引き数とオプションについて詳しくは、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

新規ディレクトリーの ACL エントリーの検査

ディレクトリーの作成後に、ディレクトリーの作成対象のユーザーおよびアプリケーションが適切な許可を受けているかどうかを確認したい場合があります。これを行うには、そのディレクトリーで **acl show** コマンドを使用して、関連する ACL エントリーを表示します。たとえば、以下のとおりです。

```
dcecp>  acl show   /.:/sales
{unauthenticated r--t-}
{group subsys/dce/cds-admin rwdtc}
{group subsys/dce/cds-server rwdtc}
{any_other r--t-}
dcecp>
```

(**acl show** コマンドについての詳細は、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

新規ディレクトリーの親ディレクトリーから必要な許可が継承されていない場合は、 **acl modify** コマンドを使用して、必要な ACL エントリーを作成します。たとえば、以下のとおりです。

```
dcecp> acl modify /.:/sales -add
{user cell_admin rwdtcia}
dcecp>
```

(**acl modify** コマンドの引き数とオプションについて詳しくは、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

セル・ルート・ディレクトリー上のディレクトリー・バージョンのアップグレード

セル・ルート・ディレクトリー上のディレクトリー・バージョンのアップグレードには特別な意味があります。この手順では、セル・ルート・ディレクトリーがセル内のすべての CDS サーバーにおいて複製される場合は、セル内のすべての CDS サーバーが最新バージョンにアップグレードされているものと暗黙指定されます。セル・ルート・ディレクトリーに **CDS_UpgradeTo** 属性を設定すると、サーバー・ソフトウェアは直ちにこのことを認識して、セル内のすべてのディレクトリーに **CDS_UpgradeTo** 属性を設定します。最終的に、セル内の影響を受けたすべてのディレクトリー上の **CDS_DirectoryVersion** 属性が新しい値にアップグレードされます。

ディレクトリー上のディレクトリー・バージョンのアップグレード

与えられた CDS のリリースの中の新しい機能を使用するには、ディレクトリーのディレクトリー・バージョンを明示的に更新する必要がある場合があります。これは一般に、ディレクトリーを複製するサーバーがすべて最新バージョンのソフトウェアにアップグレードされたときに古いバージョンが新しい機能を認識しなくなるために発生します。

ディレクトリー・バージョンをアップグレードするには、ディレクトリーに対する書き込み許可が必要であり、さらに以下のコマンドを使用しなければなりません。

```
dcecp> directory modify directory-name -add {CDS_UpgradeTo <v.n>} } -single
dcecp> directory synchronize directory-name
```

最終的に、このディレクトリーのレプリカを含むすべてのクリアリングハウスが **CDS_UpgradeTo** 属性の存在を検出し、該当のレプリカの **CDS_ReplicaVersion** 属性をアップグレードします。また、ディレクトリーを複製するすべてのクリアリングハウスで次のコマンドを使用することもできます。

```
dcecp> clearinghouse verify clearinghouse-name
```

このコマンドは、サーバーのバックグラウンド・スレッドを強制的に実行させるため、ジョブが終了するまで、他のタスクを自由に実行することができます。影響を受けたすべてのクリアリングハウスを確認した後、ディレクトリーの別のスカルクを実行して、最終的に **CSA_DirectoryVersion** 属性を適切な値に設定する必要があります。 **CDS_DirectoryVersion** 属性は、すべてのレプリカのすべての **CDS_ReplicaVersion** 属性値に新しい値が入るまで、アップグレードされません。

読み取り専用レプリカの作成

場合によっては、ディレクトリーの読み取り専用レプリカを作成する必要が生じることがあります。ディレクトリーの読み取り専用レプリカは、以下の目的で作成されます。

- ネットワーク中のディレクトリーに含まれている情報を配布し、他の場所のユーザーやアプリケーションがこの情報にアクセスしやすくする。
- 特にユーザーが遠方に分散しているネームスペースでの応答時間を向上させる。読み取り専用レプリカは、ディレクトリーに含まれている情報を最も頻繁に使用するユーザー・グループまたはアプリケーションの近くにある、クリアリングハウスに作成する必要があります。
- ディレクトリーのマスター・レプリカに含まれている情報のバックアップを保存する。複数のレプリカを維持していれば、個々のレプリカが一時的に失われてもサービスが中断されることはなく、また失われたレプリカを容易に回復できます。 1 つの特定サイトでしか使用されない情報を格納しているディレクトリーであっても、 1 つのサイトでのローカルな障害が両方のレプリカが同時に到達不能にならないように、他に少なくとも 1 つのクリアリングハウスに (できれば別の場所のサーバーに) 複製をつくっておく必要があります。 (CDS 情報のバックアップの手段としてのディレクトリーの複製についての詳細は、 207ページの『第17章 クラーク、サーバー、およびクリアリングハウスの管理』を参照してください。)

ディレクトリーの読み取り専用レプリカは、ユーザーによる変更からは保護されています。ユーザーは読み取り専用レプリカの情報を探索することはできますが、新しい情報を作成したり、既存の情報を変更したりすることは許可されていません。

読み取り専用レプリカは、 **directory create** コマンドの **-replica** オプションを使用して作成します。ディレクトリーにアクセスする必要はありますが、そのディレクトリーの内容を更新する必要がない (あるいは、更新することができない) ユーザーを持つ場合、クリアリングハウスにレプリカを作成する必要があります。

レプリカの作成の前に

レプリカを作成しようとする前に、複製するディレクトリーのマスター・レプリカを含んだクリアリングハウスが実行中であり、しかもそれが到達可能であることを確認してください。この条件が満たされているかどうかを検査するには、以下のステップに従います。

1. 複製を行いたいディレクトリーに対して **directory show** コマンドを使用して、ディレクトリーの属性値を表示し、**CDS_Replicas** 属性を調べます。この属性の値は、現在そのディレクトリーのレプリカを格納しているクリアリングハウスの名前を示します。たとえば、以下のとおりです。

```
dcecp> directory show /.:/sales
{RPC_ClassVersion {01 00}}
{CDS_CTS 1994-08-12-09:52:30.396-04:00I0.000/00-00-c0-f7-de-56}
{CDS_UTS 1994-08-12-09:52:31.506-04:00I0.000/00-00-c0-f7-de-56}
{CDS_ObjectUUID a37d84d0-b5dc-11cd-8ffe-0000c0f7de56}
{CDS_Replicas
 {{CH_UUID ce7ed810-b5db-11cd-8ffe-0000c0f7de56}
  {CH_Name /.../Chicago1/Chicago1_CH}
  {Replica_Type Master}
  {Tower {ncacn_ip_tcp 130.105.5.16}}
  {Tower {ncadg_ip_udp 130.105.5.16}}}}
{CDS_AllUpTo 1994-08-12-09:52:31.566-04:00I0.000/00-00-c0-f7-de-56}
{CDS_Convergence medium}
{CDS_ParentPointer
 {{Parent_UUID d034bc25-b5db-11cd-8ffe-0000c0f7de56}
  {Timeout
   {expiration 1994-08-12-09:52:30.396}
   {extension +1-00:00:00.000I0.000}}
  {myname /.../Chicago1/sales}}}
{CDS_DirectoryVersion 3.0}
{CDS_ReplicaState on}
{CDS_ReplicaType Master}
{CDS_LastSkulk 1994-08-12-09:52:31.566-04:00I0.000/00-00-c0-f7-de-56}
{CDS_LastUpdate 1994-08-12-09:52:31.506-04:00I0.000/00-00-c0-f7-de-56}
{CDS_RingPointer ce7ed810-b5db-11cd-8ffe-0000c0f7de56}
{CDS_Epoch a3df2a50-b5dc-11cd-8ffe-0000c0f7de56}
{CDS_ReplicaVersion 3.0}
dcecp>
```

2. この情報で、 **-clearinghouse** および **-replica** オプションを指定した **directory show** コマンドを使用して、マスター・レプリカを格納するクリアリングハウスからの応答を入手できることを確認します。たとえば、以下のとおりです。

```
dcecp> directory
show /.:/sales -replica -clearinghouse  /.:/Chicago1_CH
{RPC_ClassVersion {01 00}}
{CDS_CTS 1994-08-12-09:52:30.396-04:00I0.000/00-00-c0-f7-de-56}
{CDS_UTS 1994-08-12-09:52:31.506-04:00I0.000/00-00-c0-f7-de-56}
{CDS_ObjectUUID a37d84d0-b5dc-11cd-8ffe-0000c0f7de56}
{CDS_Replicas
 {{CH_UUID ce7ed810-b5db-11cd-8ffe-0000c0f7de56}
  {CH_Name /.../Chicago1/Chicago1_CH}
  {Replica_Type Master}
  {Tower {ncacn_ip_tcp 130.105.5.16}}
  {Tower {ncadg_ip_udp 130.105.5.16}}}}
{CDS_AllUpTo 1994-08-12-09:52:31.566-04:00I0.000/00-00-c0-f7-de-56}
{CDS_Convergence medium}
{CDS_ParentPointer
 {{Parent_UUID d034bc25-b5db-11cd-8ffe-0000c0f7de56}
  {Timeout
   {expiration 1994-08-12-09:52:30.396}
   {extension +1-00:00:00.000I0.000}}
  {myname /.../Chicago1/sales}}}
{CDS_DirectoryVersion 3.0}
{CDS_ReplicaState on}
{CDS_ReplicaType Master}
{CDS_LastSkulk 1994-08-12-09:52:31.566-04:00I0.000/00-00-c0-f7-de-56}
{CDS_LastUpdate 1994-08-12-09:52:31.506-04:00I0.000/00-00-c0-f7-de-56}
{CDS_RingPointer ce7ed810-b5db-11cd-8ffe-0000c0f7de56}
{CDS_Epoch a3df2a50-b5dc-11cd-8ffe-0000c0f7de56}
{CDS_ReplicaVersion 3.0}
dcecp>
```

-clearinghouse および **-replica** オプションを指定した **directory show** コマンドは、ディレクトリーのすべての属性値とそのレプリカの役割を表示します。

注: ディレクトリーの既存のレプリカ・セットの中に使用不能な読み取り専用レプリカがある場合、複製は完了できません。正常なスカルク・プロセスでは、ディレクトリーのレプリカ・セット内のすべてのレプリカが使用可能になると、直ちに複製を完了します。

レプリカの作成のための許可

レプリカを作成するには、以下の許可が必要です。

- 複製したいディレクトリーに対する管理許可
- 新しいレプリカを格納するクリアリングハウスに対する書き込み許可
- レプリカの作成を成功させるためには、 **-clearinghouse** および **-replica** を指定した **directory create** コマンドを入力したサーバー・システムのサーバー・プリンシパルは、複製したいディレクトリーに対する読み取り、書き込み、および管理の許可を持っていなければなりません。

  サーバーがサーバー許可グループ **subsys/dce/cds-servers** に含まれている場合、これらの許可はすでに与えられているはずです。疑わしい場合は、**acl check** コマンドを使用して、サーバー・プリンシパルが適切な許可を有していることを確認してください。 (**acl check** コマンドについて詳しくは、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

directory create コマンドの入力

-replica および **-clearinghouse** オプションを指定した **directory create** コマンドを使用して、ディレクトリーのレプリカを作成し、指定されたクリアリングハウスにそれを格納します。たとえば、以下のコマンドは、 **/.:/mfg** ディレクトリーのレプリカを作成し、そのレプリカを **/.:/Paris1_CH** という名前のクリアリングハウスに格納します。

```
dcecp> directory create /.:/mfg -replica -clearinghouse /.:/Paris1_CH
dcecp>
```

読み取り専用レプリカの削除

レプリカが格納されているクリアリングハウスのローカル・ユーザーが、読み取り専用レプリカに含まれている情報を必要としなくなったときには、読み取り専用レプリカの削除が必要になる場合もあります。レプリカがメンバーとなっているディレクトリーを削除する準備として、またはレプリカが格納されているクリアリングハウスを永続的に削除する前に、読み取り専用レプリカの削除が必要になる場合もあります。

レプリカの削除の許可

レプリカを削除するには、以下の許可が必要です。

- レプリカを削除したいディレクトリーに対する管理許可
- レプリカを削除するクリアリングハウスに対する書き込み許可

directory delete コマンドの入力

-replica および **-clearinghouse** オプションを指定した **directory delete** コマンドを使用して、指定されたクリアリングハウスからレプリカを削除します。たとえば、以下のコマンドは、 **/.:/eng** ディレクトリーのレプリカを **/.:/Chicago2_CH** クリアリングハウスから削除します。

```
dcecp> directory delete /.:/eng -replica -clearinghouse /.:/Chicago2_CH
dcecp>
```

注: (**directory delete** コマンドを使用して) ディレクトリーそのものを削除するだけで、そのディレクトリーのマスター・レプリカを削除することができます。(マスター・レプリカを削除する方法の詳細については、 237ページの『第21章 ネームスペースの再構成』を参照してください。)

ディレクトリーのスカルク

スカルク・オペレーションとは、ネームスペースに加えられた最新の変更を定期的に配布することです。 CDS はディレクトリーの **CDS_Convergence** 属性に割り当てられた値に従って、一定の間隔で各ディレクトリーをスカルクします。更新ができる限り早くディレクトリーのすべてのレプリカに配布されるようにするには、スケジュールされた次のスカルクによる新しい情報の配布を待つのではなく、**directory synchronize** コマンドを使用して、ディレクトリーのスカルクを開始することができます。以下に示す作業を実行するときに、このコマンドを使用することができます。

- 次のスカルクまで待ちたくない場合、ディレクトリーの内容またはレプリカ・セットに加えられた重要な更新内容を配布する
- 長い間作動不能で、オンラインにやっと戻ったばかりのサーバー上にレプリカを格納する、ディレクトリーをスカルクする

ディレクトリーのスカルクのための許可

ディレクトリーをスカルクするには、以下の許可が必要です。

- ディレクトリーに対する管理、書き込み、挿入、または削除の許可。
- **directory synchronize** コマンドを入力するサーバー・システムのサーバー・プリンシパルには、スカルクしようとするディレクトリーに対する読み取り、書き込み、および管理の許可が必要です。

  サーバーがサーバー許可グループ **subsys/dce/cds-servers** に含まれている場合、これらの許可はすでに与えられているはずです。疑わしい場合は、**acl show** コマンドを使用して、サーバー・プリンシパルが適切な許可を有していることを確認してください。 (**acl show** コマンドの引き数についての詳しくは、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

directory synchronize コマンドの入力

directory synchronize コマンドを使用して、ディレクトリー上で即時スカルクを開始します。

コマンドを入力した後、**dcecp** は、スカルクが進行中の間、 **dcecp>** のプロンプトを一時的に中断します。大きなレプリカ・セットのあるディレクトリーのスカル

クには多少時間がかかる可能性があります。エラー・メッセージなしのプロンプトが戻った場合、スカルクは成功です。プロンプトが戻る前にエラー・メッセージが表示された場合は、スカルクは失敗です。

スカルクを成功させるには、ディレクトリーのレプリカ・セット内のすべてのレプリカが到達可能でなければなりません。スカルクは特に大きなレプリカ・セットのあるディレクトリーの場合や、ディレクトリーのレプリカを格納するサーバーがかなり遠方に置かれていて、ネットワークの接続が必ずしも信頼できない場合に、失敗することがあります。

スカルクが失敗しても CDS が使用不能になるわけではありません。コンタクトできないレプリカ内の情報をスカルク・プロセスで更新することはできませんが、到達できるレプリカの情報は必ず更新されます。一時的に、一部のレプリカには最新の情報が入っていて、一部はそうでないという状態になります。スカルクが失敗すると、CDS は、セット内のすべてのレプリカが最新の変更内容によって更新されるまで、 (ディレクトリーのコンバージェンス値に基づいた間隔で) 自動的にスカルク・プロセスを繰り返します。すべてのレプリカに同じ情報が含まれたときに、CDS はスカルクが成功したとみなします。

特定のディレクトリーのスカルクで失敗が続く場合は、そのディレクトリーのマスター・レプリカを格納しているサーバー上の CDS イベントのログを調べることによって、原因を判別することができます。たとえば、以下のコマンドは、**/.:/admin** ディレクトリー上でスカルクを開始します。

```
dcecp> directory synchronize /.:/admin
dcecp>
```

CDS サーバー・クロックの同期化

ディレクトリー上でスカルク・オペレーションを実行した後、以下のようなメッセージを受け取ることがあります。

```
Server clocks are not synchronized
```

これは、サーバー・クロックが同期していないことを示しています。この場合は、サーバー・システムのシステム・クロックが本当に同期しているかどうかを調べる必要があります。同期しているにもかかわらず依然としてメッセージが出る場合は、 1 つのサーバーのシステム・クロックが間違って未来の時間にセットされ、その後でリストアされたためかもしれません。これは、クリアリングハウスに無効なタイム・スタンプ (現行時刻より 5 分以上先の未来のタイム・スタンプ) が格納されている可能性があるために、 CDS の問題を発生させます。

このような場合、システム・クロックを現在の時刻に調整してから、以下に示すコマンドを入力する必要があります。

```
dcecp> clearinghouse repair <clearinghouse-name> -timestamps
```

このコマンドは、クリアリングハウスを使用不能にして、不良タイム・スタンプを分析して修復し、クリアリングハウスのチェックポイントをディスクにとり、そしてクリアリングハウスを再び使用可能にします。このコマンドを使用するには、クリアリングハウスが置かれているサーバーに対する書き込み許可が必要です。また、このコマンドは、修復が必要なディレクトリー (およびそのオブジェクト) を複製する、すべてのクリアリングハウスについて実行する必要があります。

clearinghouse repair コマンドを実行した後は、ディレクトリーを正常にスカルクできるはずです。

ディレクトリーのコンバージェンスの変更

ディレクトリーの **CDS_Convergence** 属性に割り当てられた値は、ディレクトリーのマスター・レプリカを格納するサーバーがそのディレクトリーのレプリカ・セットのスカルクを開始する頻度を決定します。ディレクトリーのコンバージェンスは、 **high**、**medium**、または **low** の値に設定できます。

high のコンバージェンス値に設定されたディレクトリーは、少なくとも 12 時間に 1 回スカルクされます。ディレクトリーが更新されると、マスター・レプリカを格納するサーバーは、直ちに新しい情報をレプリカ・セット全体に伝えようとします。この更新伝搬が失敗すると、サーバーはディレクトリーのスカルクが 1 時間以内に始まるようにスケジュールします。この初期スカルクが失敗した場合は、成功するまで、1 時間間隔で追加のスカルクが開始されます。

medium のコンバージェンス値に設定されたディレクトリーは、少なくとも 12 時間に 1 回スカルクされます。ディレクトリーが更新されると、マスター・レプリカを格納するサーバーは、直ちに新しい情報をレプリカ・セット全体に伝えようとします。伝搬が失敗すると、サーバーは次のスカルクを待ってレプリカ・セットの同期をとろうとします。

low のコンバージェンス値に設定されたディレクトリーは、少なくとも 24 時間に 1 回スカルクされます。ディレクトリーが更新されると、マスター・レプリカを格納するサーバーは、直ちに新しい情報をレプリカ・セット全体に伝えようとします。伝搬が失敗すると、サーバーは次のスカルクを待ってレプリカ・セットの同期をとろうとします。

新たに作成されたどのディレクトリーも、その親ディレクトリーのコンバージェンス値を継承します。ネームスペースを作成すると、ルート・ディレクトリーには、自動的に **medium** のコンバージェンス値が割り当てられます。この値、または作成した後のすべての下位ディレクトリーのコンバージェンス値を変更しなければ、ルートの下に作成するディレクトリーもすべて **medium** のコンバージェンス値を持ちます。大部分のディレクトリーについては、この値を変更する必要はまったくありません。ただし、時には、ディレクトリーのコンバージェンスを **high** または **low** に設定すると便利な場合があります。

ディレクトリーのコンバージェンスを修正する前に

ディレクトリーのコンバージェンスを変更する前に、ディレクトリーの現行のコンバージェンス値を検査したい場合があります。これを行うためには、**directory show** コマンドを使用して、ディレクトリーの属性値を表示させ、**CDS_Convergence** 属性値を調べます。

ディレクトリーのコンバージェンスの変更のための許可

ディレクトリーのコンバージェンスを変更するには、そのディレクトリーに対する書き込み許可が必要です。

directory modify コマンドの入力

-change オプションを指定した **directory modify** コマンドを使用して、ディレクトリーの **CDS_Convergence** 属性に対して **high**、**medium**、または **low** の値を割り当てます。たとえば、以下のコマンドは、 **/.:/sales/us** ディレクトリーのコンバージェンス値を **high** に設定します。

```
dcecp> directory modify /.:/sales/us -change {CDS_Convergence high}
dcecp>
```

第19章 ネームスペースの構造と内容の表示

セル・ネームスペースの構造と内容の表示が必要な場合、 **dcecp** を使用できます。**dcecp** は、コマンド行インターフェースで情報を表示します。ここでは、ネームスペース情報を表示するための **dcecp** の使用法について説明します。

ディレクトリーの内容の一覧表示

DCE 制御プログラム (**dcecp**) プログラムには、そのセル・ネームスペース内のディレクトリーの下位のリストを表示することができる **directory list** コマンドが用意されています。ディレクトリーの下位 (子孫) とは、その中に存在する子ポインター、クリアリングハウス、オブジェクト・エントリー、およびソフト・リンクのすべてです。

directory list コマンドを使用するには、表示したい CDS 名に対する読み取り許可を持っていなければなりません。

すべてのディレクトリーの内容のリストを表示する場合は、内容を表示したい 1 つまたは複数のディレクトリーの名前を指定して、 **directory list** コマンドを入力します。たとえば、以下のとおりです。

```
dcecp> directory list /.:/eng
/.../eng_cell.osf.org/hosts/eng/aud-acl ¥
/.../eng_cell.osf.org/hosts/eng/aud-svc ¥
/.../eng_cell.osf.org/hosts/eng/cds-clerk ¥
/.../eng_cell.osf.org/hosts/eng/cds-server ¥
/.../eng_cell.osf.org/hosts/eng/dts-entity ¥
/.../eng_cell.osf.org/hosts/eng/profile ¥
/.../eng_cell.osf.org/hosts/eng/self ¥
/.../eng_cell.osf.org/hosts/eng/CDS_CTS ¥
/.../eng_cell.osf.org/hosts/eng/CDS_UTS ¥
dcecp>
```

デフォルトでは、**directory list** コマンドは、そのディレクトリーに含まれるオブジェクトのフルネーム (***/.../****pathname* が前についたオブジェクト名) を表示します。オブジェクトの RDN のみをリストするには、 **-simplename** オプションを指定した **directory list** コマンドを入力します。

特定の種類のディレクトリーの下位の名前のみを表示するには、 **directory list** コマンドに適切なオプションを含めます。たとえば、 ***/.:/*eng** ディレクトリーに格納されているすべてのソフト・リンクの名前を表示するには、以下のコマンドを入力します。

```
dcecp> directory list  /.:/eng/ -links
/.../eng_cell.osf.org/hosts/eng/CDS_CTS ¥
/.../eng_cell.osf.org/hosts/eng/CDS_UTS
dcecp>
```

CDS 名の属性値の表示

ネームスペース内の名前と関連する属性の任意のまたはすべての現行値 (クラークやサーバーは除く) を表示するには、 **dcecp show** オペレーションを使用します。

show オペレーションの基本構文は、以下のとおりです。

object-type **show** *object-name*

ここで *object-type* は、情報を表示したい CDS オブジェクトのタイプであり、*object-name* は、照会しようとしているオブジェクトの単純名 (すなわち、完全 CDS 名) で終わる完全なディレクトリー指定です。

show オペレーションを使用するには、表示したい名前に対する読み取り許可を持っていなければなりません。

以下の例では、**show** オペレーションは、 **/.:/Chicago2_CH** クリアリングハウスに関連した **CDS_CHDirectories** 属性の現行値が表示されます。このオペレーションによって戻される表示には属性の 2 つの値が示され、それぞれの値は 2 つの部分に分かれています。属性値の 2 つの部分は、ディレクトリーの UUID とディレクトリーの名前です。 **show** オペレーションは、これらの値を別々に表示します。各値について、まず 1 行にコロンで終わる属性名をリストし、その後にその値の 2 つの部分をリストします。

```
dcecp> clearinghouse show  /.:/Chicago2_CH
{RPC_ClassVersion
 {01 00}}
{CDS_CTS 1994-01-24-07:12:51.966-05:00I0.000/00-00-c0-f7-de-56}
{CDS_UTS 1994-02-03-07:17:35.794-05:00I0.000/00-00-c0-f7-de-56}
{CDS_ObjectUUID 0094e40e-bb43-1d43-9e0a-0000c0f7de56}
{CDS_AllUpTo 1994-02-03-09:17:06.393-05:00I0.000/00-00-c0-f7-de-56}
{CDS_DirectoryVersion 3.0}
{CDS_CHName /.../Chicago2/Chicago2_CH}
{CDS_CHLastAddress
 {Tower ncacn_ip_tcp:130.105.5.16[]}}
{CDS_CHLastAddress
 {Tower ncadg_ip_udp:130.105.5.16[]}}
{CDS_CHState on}
{CDS_CHDirectories
 {dir_uuid 00595ca5-bb46-1d43-9e0a-0000c0f7de56}
 {directory /.../Chicago2}}
{CDS_CHDirectories
 {dir_uuid 00888574-bb53-1d43-9e0a-0000c0f7de56}
 {directory /.../Chicago2/subsys}}
{CDS_CHDirectories
 {dir_uuid 0069ff14-bb55-1d43-9e0a-0000c0f7de56}
 {directory /.../Chicago2/subsys/dce}}
{CDS_CHDirectories
 {dir_uuid 0023cc38-bb56-1d43-9e0a-0000c0f7de56}
 {directory /.../Chicago2/subsys/dce/sec}}
{CDS_CHDirectories
 {dir_uuid 0026d57c-bb57-1d43-9e0a-0000c0f7de56}
 {directory /.../Chicago2/hosts}}
{CDS_ReplicaVersion 3.0}
{CDS_NSCellname /.../Chicago2}
dcecp>
```

以下の例で、**show** オペレーションは、 **/.:/sales** ディレクトリーに格納された、すべてのオブジェクト・エントリーを表示します。

```
dcecp> object show  /.:/sales
{CDS_CTS 1994-06-23-15:56:44.734+00:00I0.000/08-00-2b-0f-59-bf}
{CDS_UTS 1994-08-08-22:23:54.226+00:00I0.000/08-00-2b-0f-59-bf}
{CDS_ClassVersion 1.0}
dcecp>
```

以下のコマンドは、**/.:/mfg** ディレクトリーに格納された、すべてのソフト・リンクを表示します。

```
dcecp> link show /.:/mfg
{CDS_CTS 1994-06-23-15:56:44.734+00:00I0.000/08-00-2b-0f-59-bf}
{CDS_UTS 1994-08-08-22:23:54.226+00:00I0.000/08-00-2b-0f-59-bf}
{CDS_LinkTarget = /.../abc/mfg/robotics_controller1}
dcecp>
```

クラークとサーバーの属性情報の表示

セル・ネームスペース内のクラーク・エントリーとサーバー・エントリーに関連した属性の値を表示するには、 **dcecp** コマンドの **cds** および **cdsclient** を使用します。各コマンドの基本構文は、以下のとおりです。

cds show *cds-server-name*
cdsclient show *cds-client-name*

これらのコマンドを使用するには、表示したい CDS 名に対する読み取り許可が必要です。

show オペレーションの行では、クラークとサーバーの単純名にワイルドカード文字を使用することは許されません。

以下の例では、**show** オペレーションは、ローカル・クラークに関連するすべての属性の現行値を表示します。

```
dcecp> cdsclient show /.:/hosts/hostname/cds-clerk
```

戻される表示は、以下のとおりです。

```
{Creation_Time 1996-08-01-15:39:06.052+00:00I-----}
{Protocol_Errors 0}
{Authentication_Failures 0}
{Read_Operations 1088}
{Cache_Hits 928}
{Cache_Bypasses 157}
{Write_Operations 68}
{Miscellaneous_Operations
94}
```

第20章 CDS ディレクトリーを再構成するための CDS サブツリー・コマンドの使用

時には、会社の再構成またはその他の理由で、 CDS ネームスペース内の各種のディレクトリーまたはディレクトリーのサブツリーを結合あるいは再配置する必要が生じる場合があります。

たとえば、組織内のエンジニアリング・グループ **/.:/eng** が研究開発グループ **/.:/rnd** と統合され、この 2 つのグループがアプリケーションやその他のネットワーク・リソースのセットを共用し始める場合を想定してみます。このような組織変更を、これらのディレクトリーの内容をマージすることによって、ネームスペースの階層に反映させることができます。

同様に、エンジニアリング・グループが研究開発グループの下に属するようになる場合、この変更を反映させるには、**/.:/rnd/eng** という名前の空ディレクトリーを作成してから、 **/.:/eng** ディレクトリーの内容を **/.:/rnd/eng** にマージし、 **/.:/rnd** の下に **/.:/eng** を追加します。

マージと追加の手順の概要

CDS ディレクトリーをマージまたは追加するには、 DCE 制御プログラム (**dcecp**) の **directory merge** コマンドを使用します。この 2 つの手順の基本ステップは、以下のとおりです。

1. システム・プロンプトに対して **dcecp** と入力して、DCE 制御プログラムを起動します。
2. 既存のディレクトリーの 1 つを別の既存のディレクトリーとマージするかまたは追加します。これには、**directory merge** コマンドを使用してディレクトリーの下位 (オブジェクト・エントリー、ソフト・リンク、および子ディレクトリー) に関する情報を別のディレクトリーの情報と結合するか、または既存の最下位レベルのディレクトリーの下に情報を追加します。
3. ステップ 2 でマージしたソース・ディレクトリーまたはサブツリー (およびその内容) を、 **directory delete** コマンドを使用して、階層の中の古い位置から削除します。 **link create** コマンドを使用して、削除したディレクトリー情報を同じ名前の単一のソフト・リンクで置き換え、新しい位置にある情報の探索をリダイレクトします。

注: マージされたディレクトリー内にクリアリングハウス、重複名、または到達不能名が存在する場合は、特別の処理が必要です。以下の節で説明する **merge** および **append** オペレーションでは、ソースとターゲットのディレクトリーまたはサブツリーには重複名が存在せず、影響を受けるディレクトリーのマスター・レプリカを格納するクリアリングハウスは、オペレーション開始時には使用可能かつ到達可能であると想定しています。

この節で説明する **merge** と **append** のオペレーションの例は、次に示す図のネームスペースの例を基にしています。

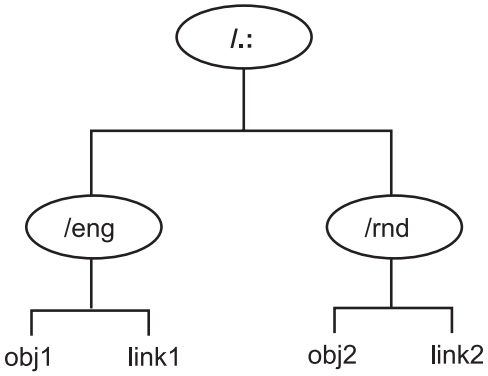


図 19. ネームスペース階層の例

このネームスペースの例は、ルートの下の **/.:/eng** および **/.:/rnd** という 2 つのディレクトリーからなります。ソース・ディレクトリー (**/.:/eng**) には、 **/.:/eng/obj1** と **/.:/eng/link1** という、2 つのエントリーが含まれています。ターゲット・ディレクトリー (**/.:/rnd**) にも、 **/.:/rnd/obj2** と **/.:/rnd/link2** という、2 つのエントリーが含まれています。

CDS ディレクトリーのマージ

以下の手順では、ソース・ディレクトリー **/.:/eng** をターゲット・ディレクトリー **/.:/rnd** にマージします。

1. **/.:/rnd** ディレクトリーとマージする前に、 **/.:/eng** ディレクトリーにスカルクを実行します。ここでソース・ディレクトリーのレプリカを同期化しておくことにより、マージ・オペレーションの失敗の原因となるエラーを防止することができます。

   ```
   dcecp> directory synchronize  /.:/eng
   dcecp>
   ```

2. **directory merge** コマンドを実行して、 **/.:/eng** ディレクトリーと **/.:/rnd** ディレクトリーをマージします。

   ```
   dcecp> directory merge /.:/eng -into  /.:/rnd
   dcecp>
   ```

   directory merge コマンドは、コマンド行引き数 (すなわち、これらのディレクトリー内のオブジェクト・エントリー、ソフト・リンク、および子ディレクトリー) で指定されている、ソース・ディレクトリーの直接の内容のみをマージすることに注意してください。

   ディレクトリーの子ディレクトリーの下位をターゲット位置にコピーするには、このコマンドの **-tree** オプションを使用しなければなりません。たとえば、前の例の **/.:/eng** ディレクトリーに **dev** と **qa** という子ディレクトリーが含まれており、これらのディレクトリーの内容をターゲット・ディレクトリー **/.:/rnd** にマージしたい場合は、以下のコマンド行を入力します。

   ```
   dcecp> directory merge /.:/eng -into /.:/rnd -tree
   dcecp>
   ```

   デフォルトでは、**directory merge** コマンドは、すべてのオブジェクト・エントリー、ソフト・リンク、および子ディレクトリーをターゲット・ディレクトリー

のマスター・クリアリングハウスに置きます。ただし、子ディレクトリーは別のクリアリングハウスに置くことができます。その場合は、このコマンドの **-clearinghouse** オプションを使用して、他のクリアリングハウスの名前を指定します。

-clearinghouse オプションには、代替クリアリングハウスを 1 つしか指定できないことに注意してください。別の代替クリアリングハウスに子ディレクトリーを置きたい場合は、各クリアリングハウスごとに **directory merge** コマンドを実行しなければなりません。あるいは、すべての子ディレクトリーを 1 つのクリアリングハウスに収める 1 つの **directory merge** コマンドを実行してから、マージ・オペレーションの後でディレクトリーを再配置しなければなりません。

注: **directory merge** コマンドで作成された CDS オブジェクトは、ソース・オブジェクトのすべての書き込み可能属性値と一部の読み取り専用属性値を保存しています。ただし、これらのオブジェクトはソース・オブジェクトの ACL を継承しません。マージされたオブジェクトがディレクトリーである場合、 **directory merge** コマンドは初期コンテナーのデフォルト ACL をそれに与えます。マージされたオブジェクトがそれ以外の CDS オブジェクト・タイプである場合は、 **directory merge** コマンドは初期オブジェクトのデフォルト ACL をそれに与えます。

directory merge コマンドは、マージ・オペレーションに問題があると、 2 つのうちのいずれかの方法で作動します。 **-nocheck** オプションを指定している場合、このコマンドはオペレーションを実行する前にエラーをチェックしません。このコマンドは、直ちにオペレーションを続行し、エラーに出会うと、停止します。 **-nocheck** オプションを省略した場合、このコマンドはマージを開始する前に、特定のエラー条件があるかどうかをチェックします。エラーを検出すると、このコマンドはエラー・メッセージを表示して停止し、それ以外の場合はマージを続行します。

directory merge コマンドによって戻されたエラー・メッセージは、問題の原因となった CDS エンティティーを識別し、その問題についての簡単な説明を示します。コマンドが問題に出会った場合は、その問題を修正してから再び実行するようにしてください。 (マージ・オペレーション中に発生する可能性のあるエラーのタイプの詳細については、 234ページの『エラーの処理』を参照してください。)

3. マージ・オペレーションが終わった後、 **/.:/eng** ディレクトリー (およびその内容) は、まだ元の位置に存在しています。元の位置から **/.:/eng** ディレクトリーを削除し、削除されたディレクトリーの位置に **/.:/eng** という名前のソフト・リンクを作成するには、以下のコマンドを実行します。このソフト・リンクは、 **obj1** および **link1** のオブジェクト・エントリーの検索を、 **/.:/rnd** ディレクトリー内のそれらの新しい位置にリダイレクトします。

   ソース・ディレクトリーを削除する前に、スカルクを実行することをお勧めします。ここでディレクトリーのレプリカを同期化すれば、削除オペレーションの失敗の原因となるエラーを防止することができます。

   /.:/eng ディレクトリーを同期化し、削除して、次に以前の内容のソフト・リンクを作成するためのコマンドの順序は、以下のとおりです。

```
dcecp> directory synchronize  /.:/eng
dcecp> directory delete  /.:/eng  -tree
dcecp> link create  /.:/eng  -to  /.:/rnd
dcecp>
```

directory delete コマンドを **-tree** オプションを指定して起動すると、ディレクトリーとそのディレクトリーの下のすべてのオブジェクト・エントリー、ソフト・リンク、および子ディレクトリーが削除されます。 **-tree** オプションを指定せずに **directory delete** コマンドを使用する場合は、削除するディレクトリーはすべて空でなければなりません。そうでないと、エラーが発生します。

図20 は、この例における、マージ・オペレーションの実行前と実行後のネームスペース例の構造を示しています。

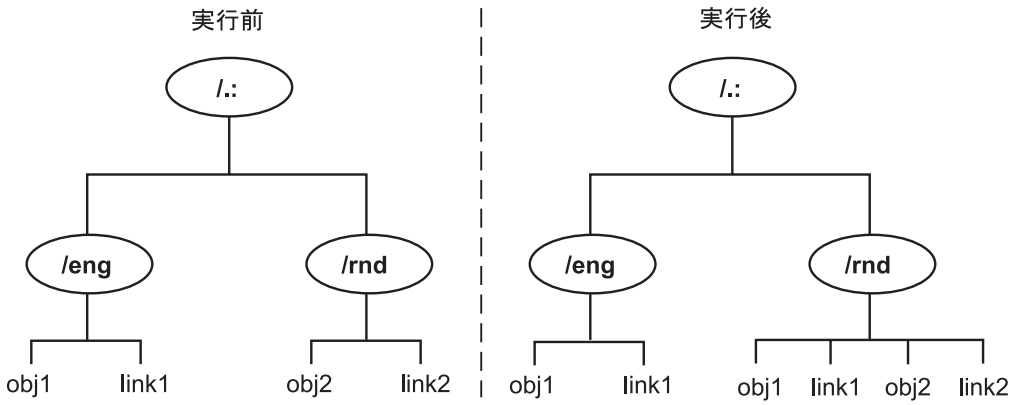


図 20. マージ・オペレーションの実行前と実行後のネームスペース例

CDS ディレクトリーの追加

以下の手順では、ソース・ディレクトリー **/.:/eng** を **/.:/rnd** ディレクトリーに追加します (すなわち、**/.:/eng** ディレクトリーを、 **/.:/rnd** ディレクトリーの下の空のターゲット・ディレクトリー **/eng** にコピーします)。

1. **directory create** コマンドを実行して、ソース・ディレクトリー **/.:/eng** の内容を置くことができる、 **/.:/rnd/eng** という名前の新しい空ディレクトリーを作成します。

   ```
   dcecp> directory create /.:/rnd/eng
   dcecp>
   ```

   デフォルトでは、**directory create** コマンドは、新しいディレクトリーを親ディレクトリーと同じクリアリングハウスに作成します。別のクリアリングハウスにディレクトリーを作成したい場合は、このコマンドの **-clearinghouse** オプションを使用して、他のクリアリングハウスの名前を指定しなければなりません。

2. **/.:/rnd** ディレクトリーに追加する前に、 **/.:/eng** ディレクトリーに対してスカルクを実行します。ここでソース・ディレクトリーのレプリカを同期化すれば、追加オペレーションの失敗の原因となるエラーを防止することができます。

   ```
   dcecp> directory synchronize  /.:/eng
   dcecp>
   ```

3. **directory merge** コマンドを実行して、ソース・ディレクトリー **/.:/eng** を **/.:/rnd** ディレクトリーに追加します (あるいは、新しい **/.:/rnd/eng** ディレクトリーにマージします)。

```
dcecp> directory merge /.:/eng -into /.:/rnd/eng
dcecp>
```

ソース・ディレクトリーに、コピーしたい内容の子ディレクトリーが含まれている場合、 **directory merge** コマンド行に **-tree** オプションを指定しなければなりません。さらに、子ディレクトリーとその内容を **/.:/rnd/eng** ディレクトリーとは別のクリアリングハウスに置きたい場合は、 **-clearinghouse** オプションを指定する必要があります。

マージ・オペレーションが成功しなかった場合は、ターゲット位置にある部分的にマージされた情報を削除してから再びこのコマンドを実行することができます。しかし、重複名がある場合は必ず削除し、影響を受けたクリアリングハウスへの接続性が確実に維持できるようにしてください。

注: directory merge コマンドで作成された CDS オブジェクトは、ソース・オブジェクトのすべての書き込み可能属性値と一部の読み取り専用属性値を保存しています。ただし、これらのオブジェクトはソース・オブジェクトの ACL を継承しません。ターゲット・オブジェクト上の ACL は、初期コンテナー (オブジェクトをマージする先の親ディレクトリー) から継承されたものか、または初期オブジェクトです。

4. 追加オペレーションが終わった後、 **/.:/eng** ディレクトリー (およびその内容) は、まだ元の位置に存在しています。元の位置から **/.:/eng** ディレクトリーを削除し、削除されたディレクトリーの位置に **/.:/eng** という名前のソフト・リンクを作成する必要があります。このソフト・リンクは、 **obj1** と **link1** のオブジェクト・エントリーの検索を、 **/.:/rnd/eng** ディレクトリー内の新しい位置にリダイレクトします。

ソース・ディレクトリーを削除する前に、スカルクを実行することをお勧めします。ここでディレクトリーのレプリカを同期化すれば、削除オペレーションの失敗の原因となるエラーを防止することができます。

/.:/eng ディレクトリーを元の位置から削除するための **dcecp** コマンドの順序は、以下のとおりです。

```
dcecp> directory synchronize  /.:/eng
dcecp> directory delete /.:/eng
dcecp> link create /.:/eng -to /.:/rnd/eng
dcecp>
```

234ページの図21 は、追加オペレーションの実行前と実行後のネームスペース例の構造を示しています。

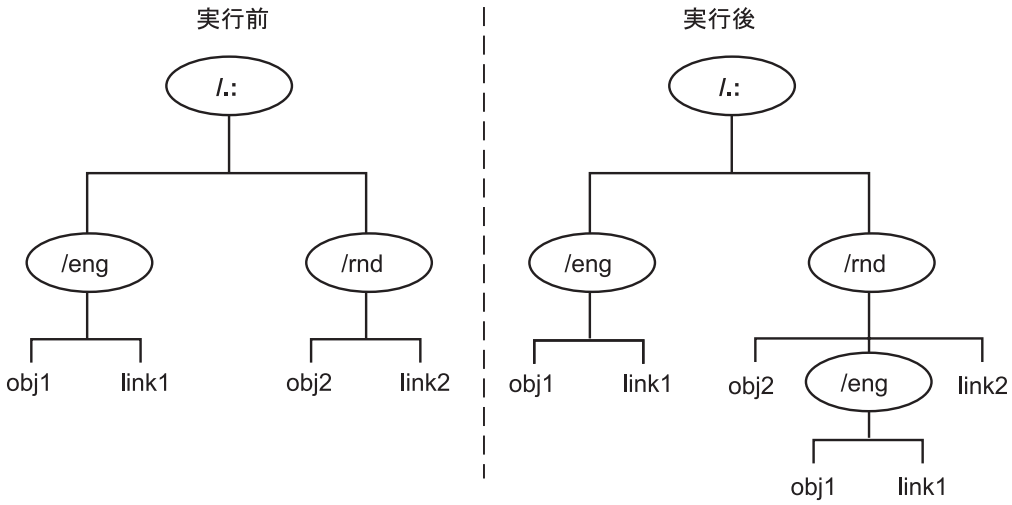


図 21. 追加オペレーションの実行前と実行後のネームスペース例

ターゲット位置での ACL の変更

ターゲット・ディレクトリー内のマージされた情報に対するプリンシパルによるアクセスを持ち続けるためには、ターゲット位置にある新しく作成されたオブジェクト上の ACL が、ソース・ディレクトリー内のオブジェクトの ACL と一致しなければなりません。 **directory merge** コマンドは、新しい位置の CDS オブジェクト上にソース ACL を再作成しないため、マージ・オペレーション後にターゲット ACL の変更が必要になる場合があります。これらの ACL を変更するには、**dcecp** の **acl replace** または **acl modify** コマンドを、 ACL 全体を置き換えたいのか、それとも ACL エントリーのみを変更したいのかに応じて使用します。

エラーの処理

directory merge コマンドがオペレーション中に出会う可能性のあるエラーの大部分は、以下のような原因によるものです。

- マージの際に検出された重複名
- コマンド実行時にマスター・クリアリングハウスが到達不能だった、ソース・サブツリー内の名前
- 許可の不足のためにターゲット位置に作成されなかったエントリー

次の各項ではこれらのエラーの回復方法について説明します。

重複名

CDS オブジェクト・エントリーまたはソフト・リンクのフルネームが、ターゲット位置にあるオブジェクト・エントリーまたはソフト・リンクのフルネームと同じである場合、 **directory merge** は、これらの重複名をリストしてから停止します。ターゲット・ディレクトリー内の同じ名前を上書きしたり、破壊したりすることを避けるために、重複名はマージされません。

重複名が存在する場合は、いずれの名前を保存するか (ソース・サブツリー内の名前かそれともターゲット・サブツリー内の名前か) を決める必要があります。決定した後、以下の方法で進めます。

1. **dcecp create** オペレーションを使用して、任意の重複オブジェクト・エントリーまたはソフト・リンクを、ソース・サブツリーまたはターゲット・サブツリー内の新しいオブジェクト・エントリーまたはソフト・リンクとして (新しい名前で) 再作成します。その後で重複名を削除します。
2. 影響を受けるクリアリングハウスへの接続性を維持できることが確かな場合は、**directory merge** コマンドを再実行して、ソース・ディレクトリーとターゲット・ディレクトリーの内容をマージします。

到達不能名の障害

時には、マージしようとしているディレクトリーのマスター・レプリカを格納したクリアリングハウスが、 **directory merge** コマンドを入力するときに使用不能または到達不能になっている場合があります。このような場合には、コマンドはディレクトリーとそれが含むエントリーを新しいターゲット位置に作成することはできません。

この理由で名前をマージできない場合は、 **directory merge** コマンドは、作成できなかった名前を示すエラー・メッセージを表示して、終了します。

不十分な許可

directory merge コマンドは、適切な許可がなければターゲット位置に CDS オブジェクトを作成することができません。コマンドが許可の不足を示すエラー・メッセージを戻した場合、ターゲット・クリアリングハウス、ディレクトリー、およびオブジェクト・エントリーの ACL を調べ、現行の許可を見て、不適切なものは変更する必要があります。

表12 は、ディレクトリーおよびその他の CDS オブジェクト・エントリーをターゲットに作成するのに必要な許可を示したものです。

表 12. ターゲット・オブジェクトの作成に必要な許可

オブジェクト	必要な許可
ディレクトリー	新規ディレクトリーのマスター・レプリカを格納するクリアリングハウスに対する書き込み許可。 新規ディレクトリーの親に対する挿入許可および読み取り許可。 新規ディレクトリーの初期コンテナーに対する挿入許可および読み取り許可。 サーバー・プリンシパルには、新しいディレクトリーの親ディレクトリーに対する読み取り許可と挿入許可も必要です。
その他の CDS オブジェクト	このオブジェクトが作成されるディレクトリーに対する挿入許可および読み取り許可。そのオブジェクト・タイプの初期オブジェクトに対する挿入許可および読み取り許可。

CDS ディレクトリーの外部セルへのマージ

directory merge コマンドを使用して、 CDS ディレクトリーを外部セルのネームスペースにマージすることもできます。一般に、この手順は、ディレクトリーやサブツリーを同じネームスペースにマージするときに使用する手順と同じです。ただし、その他にも、以下に示す、いくつか注意すべき考慮事項があります。

- 前もって、セル間認証を設定しておく必要があります。
- ソース・セルおよびターゲット・セル内のディレクトリー階層全体をマージする必要があります。

また、同一ネームスペースにマージするときと同じように、新しく作成されるターゲット・オブジェクトの ACL を変更する必要があります。

セル間認証の設定

ソース・セル内のユーザーとアプリケーションがターゲット・セル内のマージされた情報へのアクセスを正常に続けられるようにしたい場合は、ソース・セルと外部 (ターゲット) セルとの間にセル間認証の合意が存在することを確認してください。セル間認証の合意がなければ、新しくマージされた情報を要求するソース・セルからのプリンシパルは、ターゲット・セルとの通信を許可されません。セル間認証の設定方法についての詳細は、 315ページの『第6部 DCE セキュリティー・サービス』を参照してください。

外部セルへのマージ・オペレーションの実行

CDS データを外部セルのネームスペースにマージする場合は、以下のステップに従ってください。

1. 外部セル内のターゲット・マシン上の特権アカウント (**cell_admin** または **cds-admin** グループの 1 メンバー) にログインしている間に、 **directory merge** コマンドを実行して、ソース・セルのディレクトリーの内容を既存のディレクトリーにマージします。
2. ソース・セルから、マージされた情報へのアクセスを続ける場合は、ソース・サブツリーの中の一番上のディレクトリーを削除し、削除された情報をそのディレクトリーと同じ名前の 1 つのソフト・リンクで置き換えます。これで、情報の検索は外部セル内の新しい位置にリダイレクトされます。

マージされた CDS ディレクトリーのリストア

dcecp の **link delete** と **directory merge** コマンドを使用して、削除されたディレクトリーとその内容をネームスペースにリストアすることができます。

最初に **link delete** コマンドを実行して、元のソース位置にあるソフト・リンクを削除し、その後で **directory merge** コマンドを使用して、ディレクトリーのコピーを元の親ディレクトリーの下に追加します。

ディレクトリーに読み取り専用レプリカがある場合、 **directory create** コマンドを使用して、ディレクトリーが削除された各クリアリングハウスの中にディレクトリーの新しいレプリカを作成します。

directory merge コマンドは、ディレクトリーとその内容にしか作用しないことに注意してください。このコマンドは、クリアリングハウスやそれに関連するクリアリングハウス・オブジェクト・エントリーはコピーしないので、それを使用してクリアリングハウスを復元したり、異なるクリアリングハウスに常駐している個々のレプリカの間の情報の食い違いの理由を説明することはできません。さらに、特定の位置にあるディレクトリー情報が、元のマージ・オペレーション以降に変更されている可能性があります。

第21章 ネームスペースの再構成

時間の経過とともに、ネームスペース内の特定のエレメントを再構成または名前の変更を行わなければならなくなることがあります。たとえば、ユーザーに既存のネームスペース・エントリーの代替名を 1 つまたは複数提供するためのソフト・リンクを作成したい場合があります。特定のレプリカの位置とレプリカ・タイプを変更したり、あるいはディレクトリーのレプリカ・セットからレプリカを除外したりするには、このセットの再構成が必要になる場合があります。場合によっては、含まれている情報がユーザーに不要になったために、特定のディレクトリーを削除したくなる場合があります。システムで診断作業またはトラブルシューティング作業を行うため、あるいはネットワークからシステムを削除する準備のために、クリアリングハウスの再配置やサーバー・システムからのクリアリングハウスの削除が必要になる場合もあります。最後に、セルの階層を作成したり、既存の階層にセルを追加したり、あるいはセル階層の構造を変更したい場合があります。

ここでは、以下のようなネームスペースの再構成の作業の実行方法について説明します。

- ソフト・リンクの管理
- ディレクトリーのレプリカ・セットの変更
- ディレクトリーの削除
- クリアリングハウスの再配置
- クリアリングハウスの削除

ソフト・リンクの管理

ソフト・リンクとは、ネームスペース内の別の既存の名前を参照するために使用する、代替名または別名のことです。ソフト・リンクを使用すると、ユーザーとクライアント・アプリケーションは、特定のディレクトリー、オブジェクト・エントリー、またはソフト・リンクを複数の名前で参照することができます。

一般に、特定のネットワーク・リソースに代替名を割り当てる場合、またはネームスペース階層内のディレクトリーの元の名前を若干変更する場合は、ソフト・リンクを作成してください。ネームスペースを完全に設計し直す場合は、ソフト・リンクを使用すべきではありません。

ソフト・リンクの作成

DCE 制御プログラム (**dcecp**) の **link create** コマンドを使用して、ソフト・リンクを作成します。新しいソフト・リンクの名前に加えて、ソフト・リンクの宛先名、つまり、新しいソフト・リンクがポイントする既存の名前を、 **-to** オプションを用いて指定しなければなりません。宛先名には、別のソフト・リンクを含めて、ローカル・セルのネームスペース内または任意の外部セルのネームスペース内の任意の名前を指定することができます。

ソフト・リンクを作成するには、ソフト・リンクを作成しようとするディレクトリーに対する挿入許可が必要です。

注: 別のソフト・リンクをポイントするソフト・リンクを作成する場合は、ソフト・リンク・ループを作らないように注意してください。ソフト・リンク・ル

ープは、最終的に新しいソフト・リンク自体のリンク名をポイントする宛先名を指定した場合に起こります。クラークがこのエラーを検出します。

link create コマンドで作成されるすべてのソフト・リンクは、いずれも永久的であり、そのコマンドの **-timeout** オプションを使用してソフト・リンクの **CDS_LinkTimeout** 属性の期限切れの日時の値を指定しないかぎり、決して期限切れになることはありません。

期限切れの日時は、以下の形式で入力します。

yyyy-mm-dd-hh:mm:ss

たとえば、以下の値は、ソフト・リンクが 2010 年 8 月 25 日の 4:00 p.m. にまだ存在している (すなわち手動で削除されていない) 場合は、それが格納されているディレクトリーが次にスカルクされる時点で CDS が自動的に削除することを示しています。

CDS_LinkTimeout=(2010-08-25-16:00:00)

-timeout オプションを使用してソフト・リンクの **CDS_LinkTimeout** 属性の期限切れ値を指定する場合、延長値 (すでに割り当て済みの期限切れ日時に加算される時間間隔) も指定することができます。延長値は *ddd-hh:mm:ss* の形式で入力します。たとえば、**030-00:00:00** という値は、割り当てられた期限切れ日時に達したときにソフト・リンクの宛先名がまだ存在していれば、 CDS は、さらに 30 日間待ってから、スカルク時に再び宛先名の存在をチェックするということを示しています。その時に宛先名が見つからなければ、CDS はそのソフト・リンクを削除します。

次のコマンドは、**/.:/sales/eur** という名前のディレクトリーをポイントする、**/.:/sales/asia** という名前の永久ソフト・リンクを作成します。

```
dcecp> link create /.:/sales/asia -to /.:/sales/eur
dcecp>
```

次のコマンドは、 **/.:/mfg/robotics_controller01** という名前のオブジェクト・エントリーをポイントする、 **/.:/mfg/robo1** という名前のソフト・リンクを作成し、その期限切れ日時を設定します。

```
dcecp> link create /.:/mfg/robo1 -to /.:/mfg/robotics_controller01 ¥
 -timeout 2010-12-12-09:00:00
dcecp>
```

このコマンドでは、 **CDS_LinkTimeout** 属性値に指定されている期限切れ日時は、CDS がソフト・リンク **/.:/mfg/robo1** を 2010 年 12 月 12 日の 9:00 a.m. 以降の最初のスカルク時に削除するということを示しています。

次のコマンドは、 **/.:/sales/discount_stats** という名前のオブジェクト・エントリーをポイントする、 **/.:/admin/linka** という名前のソフト・リンクを作成します。

```
dcecp> link create  /.:/admin/linka -to /.:/sales/discount_stats -timeout ¥
 {2010-01-11-12:00:00 090-00:00:00}
dcecp>
```

このコマンドでは、**CDS_LinkTimeout** 属性値に指定されている期限切れ時刻は、2010 年 1 月 11 日の 12:00 p.m. 以降の最初のスカルク時に、宛先名 **/.:/sales/discount_stats** がまだ存在しているかどうかを CDS がチェックすること

を示しています。 宛先名が存在しなければ、CDS はこのソフト・リンクを削除します。宛先名がまだ存在していれば、 **CDS_LinkTimeout** 属性値 **090-00:00:00** で指定した延長時間で指定されたとおり、ソフト・リンクはさらに 90 日間だけ有効になります。この 90 日間の延長期間が過ぎると、CDS は、宛先名が削除されるまで、90 日間隔でチェックを繰り返します。

ソフト・リンクの宛先名の変更

dcecp の **link modify** コマンドを使用して、ソフト・リンクの **CDS_LinkTarget** 属性に新しい値を指定し、ソフト・リンクを現行の宛先名からネームスペース内の他の名前にリダイレクトします。

ソフト・リンクの宛先名を変更するには、そのソフト・リンクに対する書き込み許可を持っていなければなりません。たとえば、以下のコマンドは、**/.:/admin/work_disk** という名前のソフト・リンクを、現行の宛先名から新しい宛先名 **/.:/admin/work_disk03** にリダイレクトします。

```
dcecp> link modify  /.:/admin/work_disk -change {CDS_LinkTarget ¥
 /.:/admin/work_disk03}
dcecp>
```

ソフト・リンクの期限切れ値または延長値の変更

dcecp の **link modify** コマンドを使用して、ソフト・リンクの **CDS_LinkTimeout** 属性に格納されている期限切れ値および延長値に新しい値を指定します。変更したい値が一方だけの場合も、コマンドには期限切れ値と延長値の両方を指定しなければなりません。新しい値を指定するときは、元の値の指定に使用されたのと同じ形式を使用します。期限切れ値は *yyyy-mm-dd-hh:mm:ss* の形式であり、延長値は *ddd-hh:mm:ss* の形式です。

ソフト・リンクの期限切れ値または延長値を変更するには、そのソフト・リンクに対する書き込み許可を持っていなければなりません。

次のコマンドは、 **/.:/eng/link01** という名前のソフト・リンクの期限切れ値を 2010 年 12 月 31 日の 12:00 p.m. に設定します。この例では、ソフト・リンクに対して今のところ延長値は割り当てられていません。

```
dcecp> link modify /.:/eng/link01 -change {CDS_LinkTimeout ¥
 (2010-12-31-12:00:00 000-00:00:00}
dcecp>
```

次のコマンドは、**/.:/eng/link01** という名前のソフト・リンクの期限切れ値を 2010 年 12 月 31 日 の 12:00 p.m. に設定し、さらにそのソフト・リンクの延長値を 90 日に設定します。

```
dcecp> link modify /.:/eng/link01 -change {CDS_LinkTimeout ¥
 2010-12-31-12:00:00 090-00:00:00}
dcecp>
```

ソフト・リンクの削除

永久ソフト・リンクがその当初の目的以上に存続していることがわかった場合、あるいはソフト・リンクに割り当てられた期限切れ期間または延長時間に達するまで待ちたくない場合は、そのソフト・リンクを自身でネームスペースから削除することができます。

link delete コマンドを使用して、指定された名前のソフト・リンクを削除します。

ソフト・リンクを削除するには、そのソフト・リンクに対する削除許可、またはそのソフト・リンクを格納しているディレクトリーに対する管理許可がなければなりません。

たとえば、次のコマンドは、**/.:/dist/pointer_1** という名前のソフト・リンクを削除します。

```
dcecp> link delete  /.:/dist/pointer_1
dcecp>
```

ディレクトリーのレプリカ・セットの変更

ディレクトリーのレプリカ・セットには常にマスター・レプリカが含まれいますが、他の読み取り専用レプリカも含まれている場合があります。ディレクトリーと関連した **CDS_Replicas** 属性に格納されている値には、存在するレプリカの数、レプリカのタイプ、およびそれぞれのレプリカが格納されているクリアリングハウスの名前を含む、そのディレクトリーのレプリカ・セットを記述する情報が含まれています。 **dcescp directory modify** コマンドを使用して、ディレクトリーの **CDS_Replicas** 属性に格納されている現行値を上書きし、以下のタスクの一方または両方を 1 つのコマンドで実行することができます。

- ディレクトリーのレプリカ・セット内に新しいマスター・レプリカを指定する。
- レプリカをディレクトリーのレプリカ・セットから除外する。

注: **directory modify** コマンドの一部として、CDS は、ディレクトリーで即時スカルクを開始して、レプリカ・セットのすべてのメンバーに対して変更内容をできる限り早く配布することができます。

レプリカ・セットを変更する前に

ディレクトリーのレプリカ・セットを変更する前に、存在するレプリカの数、レプリカのタイプ、およびそれぞれのレプリカが格納されているクリアリングハウスの名前を知っていなければなりません。ディレクトリーのレプリカ・セットの変更に使用されるコマンドは、レプリカが新しいセットから誤って抜けてしまうことがないようになっています。セット内のすべての既存のレプリカを明示的にリストする必要があります。新しいセットに任意のレプリカを含めたり除外したりすることができますが、すべてのレプリカを明らかにしなければなりません。新しいセットに含めるレプリカの中でマスター・レプリカとして指定できるのは 1 つだけです。

ディレクトリーのすべてのレプリカの名前を表示するには、 **dcecp** の **directory show** コマンドを使用します。このコマンドは、この情報を収集するためにディレクトリーの **CDS_Replicas** 属性を照会します。 (**dcecp directory show** コマンドの使用法については、 215ページの『第18章 CDS ディレクトリーの管理』を参照してください。)

レプリカ・セットの変更に必要な許可

ディレクトリーのレプリカ・セットを変更するための許可は、以下のとおりです。

- ディレクトリーに対する管理許可が必要です。また、サーバー・プリンシパルにはディレクトリーに対する管理、読み取り、および書き込み許可が必要です。

- 新しいマスター・レプリカを指定するときは、現行マスター・レプリカを格納するクリアリングハウスに対する書き込み許可も持っていなければなりません。サーバー・プリンシパルには、新しいマスター・レプリカとして指定する予定の読み取り専用レプリカを格納している、クリアリングハウスに対する書き込み許可が必要です。

  新しいマスター・レプリカが置かれるサーバー上のサーバー・プリンシパルには、ディレクトリーに対する管理、 読み取り、および書き込みの許可が必要です。

どのレプリカを含め、どのレプリカを除外するかを知っており、変更が必要な許可をすでに変更してある場合は、 **directory modify** コマンドを実行して、ディレクトリーのレプリカ・セットを変更します。 2 つのオプション (新しいマスター・レプリカの指定、および既存の読み取り専用レプリカの除外) についての説明は、以下の節で行います。

新しいマスター・レプリカの指定

時として、構成管理のために、異なったレプリカをディレクトリーのマスター・レプリカとして指定したい場合があります。

たとえば、以下の場合は、新しいマスター・レプリカを指定することができます。

- そのクリアリングハウスに 1 つまたは複数のマスター・レプリカが入っているサーバー・システムが、長い間ダウンするか、または永久にネットワークから削除される場合。
- 1 つまたは複数のマスター・レプリカを格納するクリアリングハウスが、ネームスペースから削除される場合。
- ディレクトリーに対する更新の大部分が発生する場所の近くに、マスター・レプリカを置きたい場合。

新しいマスター・レプリカを指定するには、**dcecp directory modify** コマンドを使用します。

図22 は、レプリカ・セットの例を示したものです。 **/.:/eng** ディレクトリーのこのレプリカ・セットは、 3 つのレプリカ、つまり、クリアリングハウス **/.:/NY1_CH** に格納されたマスター・レプリカ、クリアリングハウス **/.:/NY2_CH** に格納された読み取り専用レプリカ、およびクリアリングハウス **/.:/Chicago1_CH** に格納された読み取り専用レプリカからなります。

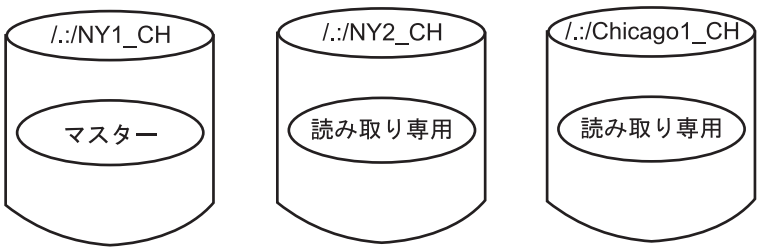


図 22. レプリカ・セットの例

次のコマンドは、クリアリングハウス **/.:/Chicago1_CH** に格納されている読み取り専用レプリカをディレクトリーの新しいマスター・レプリカとして指定し、前のマ

スター・レプリカ (クリアリングハウス **/.:/NY1_CH** に格納されている) を読み取り専用レプリカとして指定し、そして、クリアリングハウス **/.:/NY2_CH** に格納されている読み取り専用レプリカはそのままにしておきます。

```
dcecp> directory modify /.:/eng -master /.:/Chicago_1_CH ¥
> -readonly {/.:/NY1_CH /.:/NY2_CH}
dcecp>
```

図23 は、上記のコマンドの結果を示したものです。

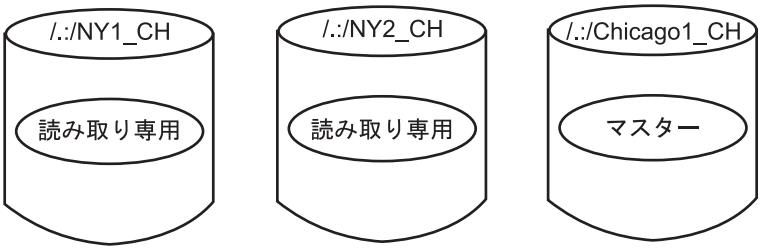


図 23. マスター再指定後のレプリカ・セットの例

レプリカ・セットからのレプリカの除外

レプリカが格納されているクリアリングハウスが突然使用不能になった時に、レプリカ・セットからレプリカを一時的に除外することができます。こうすると、除外されたレプリカが使用不能な間に、 CDS がディレクトリーのスカルクを完了できるようになります。

レプリカ・セットからレプリカを除外する場合には、 **exclude** 引き数を指定した **dcecp directory modify** コマンドを使用して、指定したレプリカを除外してディレクトリーのレプリカ・セットを再作成します。コマンドの中で既存のレプリカをすべて明らかにしなければならないことを覚えておいてください。

以下の例で、**/.:/eng** ディレクトリーのレプリカ・セットは、3 つのレプリカ、つまり、クリアリングハウス **/.:/Chicago1_CH** に格納されたマスター・レプリカ、およびクリアリングハウス **/.:/NY1_CH** と **/.:/NY2_CH** に格納された読み取り専用レプリカからなります。

この場合、**/.:/NY1_CH** クリアリングハウスは、ネットワーク伝送回線の偶発的損傷のために、ネットワークから切り離されます。クリアリングハウスへの接続性は数日間復元されません。この期間、クリアリングハウス **/.:/NY1_CH** に格納されている読み取り専用レプリカを一時的に除外しない限り、 **/.:/eng** ディレクトリーのスカルクは失敗します。

修復期間中でも **/.:/eng** ディレクトリーのスカルクが正常に実行されるようにするには、以下のコマンドを入力して、**/.:/eng** ディレクトリーの **CDS_Replicas** 属性の現行値を、 **/.:/NY2_CH** と **/.:/Chicago1_CH** のクリアリングハウスに格納されているレプリカのみを含む、新しい値で上書きします。

```
dcecp> directory modify/.:/eng -master /.:/Chicago1_CH ¥
> -readonly /.:/NY2_CH -exclude /.:/NY1_CH
dcecp>
```

図24 は、上記のコマンドの結果を示したものです。

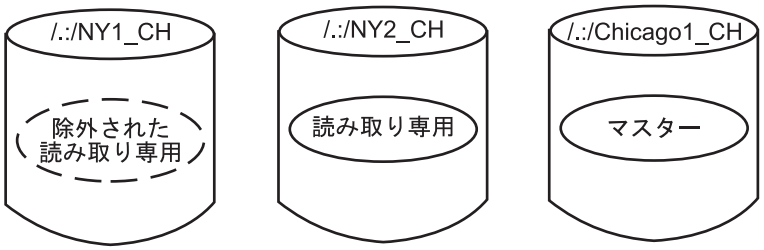


図 24. レプリカ除外後のレプリカ・セットの例

/.:/NY1_CH のクリアリングハウスとの接続性が再び確立されたら、以下のコマンドを入力して、クリアリングハウス **/.:/NY1_CH** に格納されている読み取り専用レプリカをレプリカ・セットに再インストールします。

```
dcecp> directory modify /.:/eng -master /.:/Chicago1_CH ¥
> -readonly {/.:/NY1_CH /.:/NY2_CH}
dcecp>
```

注: 除外されたレプリカが常駐するクリアリングハウスが再度使用可能になった時は常に、できるだけ早くそれらのレプリカをレプリカ・セットに再インストールしてください。

ディレクトリーの削除

場合によっては、ディレクトリーに含まれている情報をユーザーが必要としなくなったときに、ネームスペースからそのディレクトリーを削除したい場合があります。ディレクトリーを削除するときには、次の 2 つのことを考慮しなければなりません。

- そのディレクトリーに子ディレクトリーまたは他の CDS オブジェクトのエントリーが含まれているか? ディレクトリーを削除するには、それが空でなければなりません。
- そのディレクトリーのレプリカがあるか? それぞれのレプリカは別々に削除しなければなりません。

これらの考慮事項について、以下の節で説明します。

ディレクトリーを削除するには、以下の許可が必要です。

- ディレクトリーに対する削除許可。
- ディレクトリーのマスター・レプリカを格納するクリアリングハウスに対する書き込み許可。
- **directory delete** コマンドを入力するサーバーのサーバー・プリンシパルには、親ディレクトリーに対する管理許可、または削除しようとするディレクトリーをポイントする子ポインターに対する削除許可が必要です。

  サーバーがサーバー許可グループ **subsys/dce/cds-servers** に含まれている場合、これらの許可はすでに与えられているはずです。疑わしい場合は、**dcecp** ユーティリティーの **acl show** を使用して、サーバー・プリンシパルが適切な許可

を持っていることを確認してください。 (**acl show** コマンドについての詳細は、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

非複製ディレクトリーの削除

レプリカを持たないディレクトリーを削除するには、 **dcecp** の **directory delete** コマンドを使用します。たとえば、ディレクトリー **/.:/sales**、それ自体のすべての内容、およびそのいずれかの子ディレクトリーの内容を削除するには、以下のように入力します。

```
dcecp> directory delete /.:/sales -tree
dcecp>
```

注: **directory delete** コマンドの -tree オプションを使用する場合は注意してください。このコマンドは、コマンド行で指定したディレクトリーを削除したいかどうかを確認せずに、直ちに削除オペレーションに進んでしまいます。この結果、残しておきたいディレクトリーが失われる恐れがあります。

dcecp コマンドの働きは、スクリプトを使用して変更できることに注意してください。 **directory delete** コマンドの場合、 **-tree** オプションを指定してコマンドを実行したときに削除オペレーションを確認するプロンプトを必ず出すスクリプトを作成することができます。スクリプトの作成についての説明は、本書の 1ページの『第1部 DCE 制御プログラム』を参照してください。

ディレクトリーとそのエントリーを誤って削除するのを防ぐための 1 つ方法は、**directory delete** コマンドを実行する前に内容を表示することです。 CDS ディレクトリーの内容をエントリー・タイプごとに表示するには、 **-object**、**-link**、および **-directory** の各オプションを指定した **directory list** コマンドを使用します。

次に示すのは、**/.:/sales** という名前のディレクトリーを削除する例です。このディレクトリーには、1 つのオブジェクト・エントリーと 1 つのソフト・リンクがあります。

```
dcecp> directory list /.:/sales -simplename work_disk link1
dcecp> directory list /.:/sales -simplename -object work_disk
dcecp> directory list /.:/sales -simplename -link link1
dcecp> directory delete /.:/sales -tree
dcecp> directory show /.:/sales
Error: Requested entry does not exist
dcecp>
```

削除しようとするディレクトリーが空ではない場合、 **directory delete** コマンドは失敗します。この種の障害から回復するには、ディレクトリーおよびその子ディレクトリー内のすべてのエントリーを削除した後で、再び **directory delete** コマンドを実行します。 **link delete** コマンドおよび **object delete** コマンドを使用して、任意のディレクトリー内のソフト・リンクとオブジェクト・エントリーを削除します。次に、**directory delete** コマンドを使用して、ディレクトリーを削除します。

ディレクトリー・レプリカの削除

ディレクトリーが複製されている場合は、すべてのレプリカを個別に削除しなければなりません。そうすれば、前の節で説明したコマンドを使用してディレクトリーを削除することができます。

ディレクトリーのすべてのレプリカのリストを表示するには、 **dcecp directory show** コマンドを使用します。リスト内のディレクトリーの **CDS_Replicas** 属性を見てください。各レプリカの **CDS_Replicas** 属性には、複数の副属性があります。レプリカが置かれているクリアリングハウスの名前を入手するには、各レプリカの **CH_Name** 副属性を調べてください。たとえば、以下のとおりです。

```
dcecp> directory show /.:/sales
{RPC_ClassVersion {01 00}}
{CDS_CTS 2010-05-06-11:41:05.314-05:00I0.000/08-00-09-25-13-52}
{CDS_UTS 2010-06-21-03:06:08.842-05:00I0.000/08-00-09-25-13-52}
{CDS_ObjectUUID 5f97a584-bf9b-11cd-9362-080009251352}
{CDS_Replicas
 {{CH_UUID de3401e6-bb98-11cd-aac5-080009251352}
  {CH_Name /.../absolut_cell/absolut_ch}
  {Replica_Type Master}
  {Tower {ncacn_ip_tcp 130.105.5.93}}
  {Tower {ncadg_ip_udp 130.105.5.93}}}}
{CDS_AllUpTo 23854-01-29-19:45:44.841-05:00I0.000/08-00-09-25-13-52}
{CDS_Convergence medium}
{CDS_ParentPointer
 {{Parent_UUID df13b228-bb98-11cd-aac5-080009251352}
  {Timeout
   {expiration 2010-08-24-19:30:30.827}
   {extension +1-00:00:00.000I0.000}}
  {myname /.../absolut_cell/sales}}}
{CDS_DirectoryVersion 3.0}
{CDS_ReplicaState on}
{CDS_ReplicaType Master}
{CDS_LastSkulk 2010-01-29-19:45:44.841-05:00I0.000/08-00-09-25-13-52}
{CDS_LastUpdate 2010-06-21-03:06:08.842-05:00I0.000/08-00-09-25-13-52}
{CDS_Epoch 60ac0730-bf9b-11cd-9362-080009251352}
{CDS_ReplicaVersion 3.0}
dcecp>
```

ディレクトリーの名前とクリアリングハウスの名前を使用すると、各レプリカを固有に識別することができます。レプリカを削除するには、一連の **directory delete** コマンドの中でこれらの名前を使用します。各レプリカの名前はコマンドの引き数であり、クリアリングハウスの名前を **-clearinghouse** オプションの値として使用してください。削除するディレクトリーがレプリカであることを示すために、**-replica** オプションがコマンド行にもなければなりません。コマンド行の例は、以下のとおりです。

```
dcecp> directory delete /.:/sales -replica -clearinghouse /.:/NY1_CH
dcecp>
```

注: **directory delete** コマンドでは、ディレクトリー・レプリカを処理するために、それが空である必要はありません。このコマンドはレプリカ、そのすべての内容、およびその子ディレクトリーを、オペレーションの確認のプロンプトを出さずに、直ちに削除します。

CDS_Replicas 属性を調べ、すべてのレプリカを検出し、そして、それらを 1 つのコマンドで削除する、**dcecp** スクリプトを作成することもできます。スクリプトの作成についての説明は、本書の 1ページの『第1部 DCE 制御プログラム』を参照してください。

クリアリングハウスの再配置

注: この節では、クリアリングハウスをある CDS サーバー・システムから別の CDS サーバー・システムに一時的に再配置するプロシージャーについて説明します。このプロシージャーは、追加の CDS サーバー・システムの構成には使用できないので注意してください。 (CDS サーバーと CDS クラークの構成方法については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド ― 入門* を参照してください。)

時には、クリアリングハウスを、現在常駐しているサーバー・システムから別のサーバー・システムに再配置する必要性が生じる場合があります。たとえば、クリアリングハウスを移動したくなる場合は以下のような場合です。

- ホスト・サーバー・システムを、修理またはその他の理由で、ネットワークから一時的に切断する必要がある場合。
- 現行ホスト・システムを CDS サーバーとして使用する必要がなくなった場合。
- ネットワーク上で物理的に、クリアリングハウスに含まれている情報を使用するユーザー・グループやアプリケーションの近くにある、サーバー・システムにクリアリングハウスを移動したい場合。

クリアリングハウスを再配置するには、以下のステップに従います。

1. CDS サーバー A および B に cell_admin としてログインする。
2. サーバー A に次のコマンドを入力する。

   ```
   dcecp>  dcecp -c clearinghouse
   disable /.:/my_ch
   ```

3. サーバー A のディレクトリーを **/opt/dcelocal/var/directory/cds** に変更する。
4. **/.:/my_ch** に関連付けられているエントリーを、 **cdsfiles.map** ファイルから探す。このようなエントリーは、次のようになっています。

   ```
   /opt/dcelocal/var/directory/cds/acell#my_ch.checkpoint001
   /opt/dcelocal/var/directory/cds/acell#my_ch.tlog001
   /opt/dcelocal/var/directory/cds/acell#my_ch.version
   ```

5. ステップ 4 で検出したファイルを、サーバー B の同じディレクトリーに移す。
6. ステップ 4 でサーバー A の **cdsfiles.map** ファイルから検出したマップ・エントリーを抽出し、サーバー B の **cdsfiles.map** ファイルに追加する。
7. サーバー A 上で、ステップ 4 でサーバー A の **cdsfiles.map** ファイルから検出したエントリーと、それに関連付けられているファイルとを削除する。
8. サーバー B に次のコマンドを入力する。

   ```
   dcecp>  dcecp -c clearinghouse create /.:/my_ch
   ```

9. サーバー B に次のコマンドを入力する。

   ```
   dcecp>  dcecp -c cdscache discard
   ```

クリアリングハウスとそのホスト・サーバー・システムとの分離

CDS サーバーが開始するときには必ず、サーバー・ソフトウェアが実行するタスクの 1 つが、その 1 つまたは複数のクリアリングハウスを開始します。サーバーは、システムに常駐のクリアリングハウスのリストを調べることによって、このタスクを自動的に実行します。クリアリングハウスを再配置する前に、 **dcecp** の

clearinghouse disable コマンドを使用して、クリアリングハウス・ファイルを更新し、ファイルが確実に整合するようにしてから、ターゲット・サーバーにコピーします。

clearinghouse disable コマンドは、さらに、指定されたクリアリングハウスの知識をソース・サーバーの内部メモリーから削除します。これによって、再配置されたクリアリングハウスが、サーバーの再起動時に、ソース・サーバーで自動的に起動されることがなくなります。

clearinghouse disable コマンドを使用するには、クリアリングハウスが置かれているサーバーに対する書き込み許可が必要です。

以下のコマンド例は、クリアリングハウス **/.:/Chicago2_CH** の知識をそのホスト・サーバーのメモリーから削除します。

```
dcecp> clearinghouse disable /.:/Chicago2_CH
dcecp>
```

ターゲット・サーバー・システムへのクリアリングハウス・データベース・ファイルのコピー

クリアリングハウスを使用不能にしてクリアリングハウスの情報をホスト・サーバーから削除した後で、クリアリングハウス・データベース・ファイルを新しいホスト・サーバー・システム上の特定の位置にコピーしなければなりません。

クリアリングハウス・データベースは、以下の 3 つのファイルで構成されます。

- *clearinghouse-name*.**checkpoint***nnnnnnnn*
- *clearinghouse-name*.**tlog***nnnnnnnn*
- *clearinghouse-name*.**version**

ここで、*nnnnnnnn* はクリアリングハウス名を表します。

ファイルを新しいホスト・システムにコピーする前に、これらのファイルが存在することを確認しなければなりません。 (すべての CDS ファイルの完全パス名については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド − 入門* と *OSF DCE Porting and Testing Guide* を参照してください。)

注: ディレクトリーの中に、 2 つの **.checkpoint***nnnnnnnn* ファイルが見つかることが時々あります。これは、クリアリングハウスの最新のチェックポイント・オペレーション中のシステム・クラッシュあるいはその他の中断の結果として起こる場合があります。 2 つのファイルが見つかった場合は、その両方をターゲット・サーバー・システムにコピーしてください。ターゲット・サーバーでクリアリングハウスが使用可能になると、すぐにそのシステム上のサーバー・ソフトウェアが自動的に問題を調整します。

データベース・ファイルを新しい CDS サーバーに移動するには、 **ftp** ユーティリティー、または類似のネットワーク・ファイル転送ユーティリティーを使用します。既存のサーバー・ホストから新しい CDS サーバー・ホストに、この 3 つのデータベース・ファイルをコピーしてください。新旧の CDS サーバーにファイルが置かれているディレクトリーは、 *dcelocal***/var/directory/cds** です。

ターゲット・サーバーでのクリアリングハウスの開始

クリアリングハウス・データベース・ファイルをターゲット・サーバー・システムの適切な位置にコピーした後、 **clearinghouse create** コマンドを使用して、クリアリングハウスを新しい場所で開始します。元の (ソース) 位置で使用されていたのと同じクリアリングハウス名を指定していることを確認してください。コマンドを入力後、サーバーは、クリアリングハウスのファイルを検出し、その知識をメモリーに追加してから、クリアリングハウスを開始します。

クリアリングハウスの再配置のために **clearinghouse create** コマンドを使用するには、このクリアリングハウスを再配置しようとするサーバーに対する書き込み許可が必要です。

上記の例では、クリアリングハウス **/.:/Chicago2_CH** のデータベース・ファイルは、 **orion** という名前のサーバー・システムに正常にコピーされています。 **orion** で出される以下のコマンドは、 **/.:/Chicago2_CH** という名前のクリアリングハウスをそのサーバーに再配置します。

```
dcecp> clearinghouse create /.:/Chicago2_CH
dcecp>
```

クリアリングハウスの削除

以下の場合には、クリアリングハウスが置かれているサーバー・システムからこのクリアリングハウスを削除する必要があります。

- システムの再割り振りまたはネットワークからの削除がスケジュールされている場合。
- システムを CDS サーバーとして使用する必要がなくなった場合。

クリアリングハウスを削除する前に

クリアリングハウスの削除を行う前に、次のことを確認してください。

- クリアリングハウスをサーバーが認識している。
- クリアリングハウスにマスター・レプリカが格納されていない。

クリアリングハウスをクリアした場合、クリアリングハウスが動作しているサーバーの内部メモリーには、このクリアリングハウスに関する情報はもうありません。後でクリアリングハウスを削除しようとしても、 CDS はこのクリアリングハウスを検出できず、このクリアリングハウスが存在しないというメッセージを戻します。クリアされたクリアリングハウスを削除するには、その前に、**clearinghouse create** コマンドを使用して、そのクリアリングハウスを再作成しなければなりません。

CDS では、ディレクトリーのマスター・レプリカが入っているクリアリングハウスを削除することはできません。このようなクリアリングハウスを削除するには、その前に、そのディレクトリーのレプリカ・セット内の別のレプリカをマスター・レプリカとして指定しておかなければなりません。ディレクトリーのレプリカがそれ以上ない場合は、別のクリアリングハウスで読み取り専用レプリカを作成して、ディレクトリーの新しいマスター・レプリカとして指定してから、元のマスター・レ

プリカをクリアリングハウスから削除します。 (ディレクトリーのレプリカ・セットの変更についての説明は、 240ページの『ディレクトリーのレプリカ・セットの変更』を参照してください。)

クリアリングハウスの削除の許可

クリアリングハウスの削除には、以下の許可が必要です。

- クリアリングハウスに対する書き込み許可と削除許可、およびクリアリングハウスにレプリカを格納しているすべてのディレクトリーに対する管理許可。
- サーバー・プリンシパルには、関連するクリアリングハウス・オブジェクト・エントリーに対する削除許可、およびクリアリングハウスにレプリカを格納しているすべてのディレクトリーに対する管理許可が必要です。

クリアリングハウスの削除

clearinghouse delete コマンドを使用して、クリアリングハウスを削除します。また、このコマンドは、クリアリングハウスに関連するクリアリングハウス・オブジェクト・エントリーおよびすべての読み取り専用レプリカもクリアリングハウスから削除します。

クリアリングハウスの削除を完了するにはしばらく時間がかかる場合があります。CDS がクリアリングハウスを削除するのは、クリアリングハウスに読み取り専用レプリカを格納したすべてのディレクトリーのスカルクが正常に完了した後です。

以下のコマンドの例は、**/.:/Paris2_CH** クリアリングハウスを削除します。

```
dcecp> clearinghouse delete /.:/Paris2_CH
dcecp>
```

第22章 セル間ネーミングの管理

ローカル・セル外の名前を検索するには、 CDS クラークが他のセルのディレクトリー・サーバーを見つける方法を持っていなければなりません。グローバル・ディレクトリー・エージェント (GDA) は、グローバル・ネーミング環境を介して他のセルへの接続の役割を果たし、セル間の通信を可能にします。ここでは、GDA の動作方法およびその管理方法について説明します。この章では、ローカル・セルを他のセルにアクセスできるようにするために必要なステップである、 X.500、LDAP、または DNS のいずれかにローカル・セルを定義する方法も説明します。

注: セル名が X.500 形式の名前である場合は、 GDS または LDAP サーバーをグローバル名サーバーとして使用できます。

グローバル・ディレクトリー・エージェントの働き

GDA は、ローカル・セル内の CDS クラークと他のセル内の CDS サーバーとの間の仲介者です。 CDS クラークは、GDA を他の名前サーバーと同ように扱って、名前検索要求をこれに渡します。ただし、GDA はクラークに特定のサービスを 1 つしか提供しません。すなわち、X.500、 LDAP、または DNS ネームスペース内のセル名を検索して、その結果をクラークに戻します。するとクラークは、その結果を使用して外部セル内の CDS サーバーと連絡します。

GDA は、他のセルとの通信を希望する、すべてのセル内に存在していなければなりません。 これは CDS サーバーと同じシステム上にあってもよく、また他のシステム上に独立して存在することもできます。可用性と信頼性の向上のために、 1 つのセルの中に複数の GDA を構成することができます。 CDS サーバーと同様、 GDA はプリンシパルであり、それ自体をクラークに対して認証しなければなりません。

CDS は、セル・ルート・ディレクトリーに関連した **CDS_GDAPointers** 属性に格納されているアドレス情報を読み、GDA を見つけます。 GDA プロセスは、開始すると必ず、新しいエントリーを作成するかまたはルート・ディレクトリーの **CDS_GDAPointers** 属性内の既存のエントリーを更新します。このエントリーには、現在 GDA が実行されているホストのアドレスが入っています。 1 つのセル内に複数の GDA が存在する場合、それらは各自のアドレス情報を **CDS_GDAPointers** 属性内に作成し、保守します。

CDS サーバーがローカル・セルにない名前に対する要求を受け取ると、サーバーはセル・ルート・ディレクトリーの **CDS_GDAPointers** 属性を調べて、 1 つまたは複数の GDA の位置を検索します。 252ページの図25 は、 CDS クラークと CDS サーバーが GDA の検索のためにどのように相互作用するかを示しています。

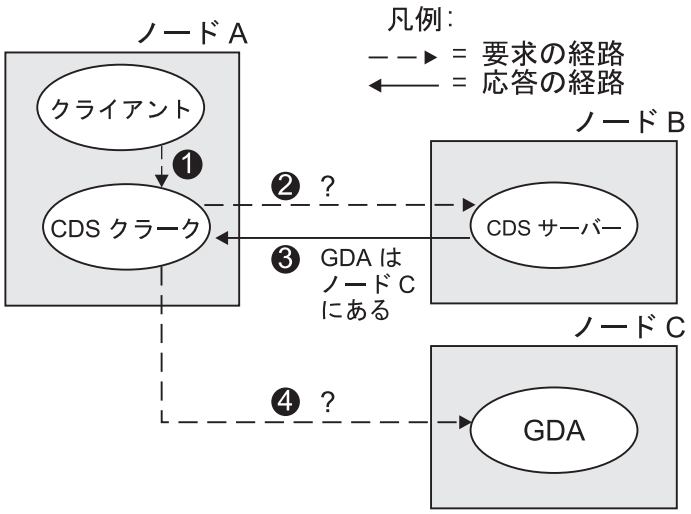


図 25. CDS クラークが GDA を検索する方法

次のステップは、上記の図で示されている GDA 検索をまとめたものです。

1. ノード A で、クライアント・アプリケーションが **/...** 接頭部で始まるグローバル名を CDS クラークに渡します。
2. クラークは、ノード B 上で認識している CDS サーバーに検索要求を渡します。
3. サーバーのクリアリングハウスにはセル・ルート・ディレクトリーのレプリカが含まれているので、サーバーは **CDS_GDAPointers** 属性を読んで、 GDA が実行されているノード C のアドレスを戻します。
4. クラークは、検索要求を GDA に渡します。

253ページの図26 は、 DNS 内で定義されている外部セル内の名前を見つけるために CDS と GDA がどのように相互作用するかを示しています。名前が外部セル内の印刷サーバーを表す **/.../widget.com/printsrv1** であるとします。

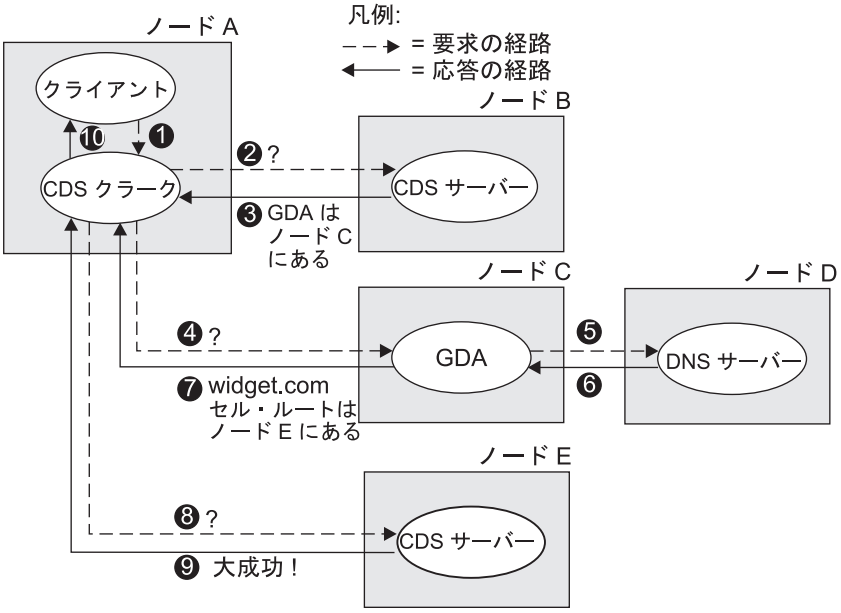


図 26. GDA が CDS の名前検索を支援する方法

次のステップは、上記の図で示されている名前の検索をまとめたものです。

1. クライアント・アプリケーションが名前 **/.../widget.com/printsrv1** を CDS クラークに渡します。
2. クラークは、ノード B 上で認識している CDS サーバーに検索要求を渡します。
3. サーバーのクリアリングハウスにはセル・ルート・ディレクトリーのレプリカが含まれているので、サーバーは **CDS_GDAPointers** 属性を検索し、 GDA が実行されているノード C のアドレスを戻します。
4. クラークは、検索要求を GDA に渡します。
5. GDA は、名前が DNS スタイル名であることを認識し、 2 番目のコンポーネントが DNS に定義されているセル名であると想定します。そして、名前のその部分 (**widget.com**) を DNS に渡します。簡単にするために、この図では DNS サーバーを 1 つだけ示してあります。実際には、複数の DNS サーバーがグローバル・セル名の解決に関係している場合があります。

   注: これは DNS スタイルの名前の探索例ですが、オペレーションの順序と実行方法は X.500 名または階層セル名の場合もほとんど同じです。 GDA は、名前が X.500 スタイル名であると認識すると、その名前を、DNS サーバーに渡すのではなく、LDAP クライアント (LDAP API を介して)、または GDS クライアント (XDS/XOM API を介して) に渡します。 LDAP クライアントまたは GDS クライアントは、該当するサーバーと通信して、セル・バインディング (DNS サーバーから入手するのと同じ情報) を入手します。
6. DNS は検索を行って、 **widget.com** セル・エントリーに関連した情報を GDA に戻します。この情報には、 **/.../widget.com** セル・ネームスペースのルート・ディレクトリーのレプリカを維持している、サーバーのアドレスが含まれます。

7. GDA は外部セルに関する情報をクラークに渡します。
8. クラークは外部セル内のノード E 上の CDS サーバーと連絡し、それに検索要求を渡します。
9. ノード E のサーバーのクリアリングハウスにはルート・ディレクトリーのレプリカが含まれているので、このサーバーはルートの中で **printsrv1** のエントリーを検索し、要求された情報をノード A のクラークに渡します。簡単にするために、この例では、クラークは外部セル内の 1 つのサーバーとのみコンタクトしています。フルネームの解決時に、クラークは実際には外部セル内の複数のサーバーに対する照会を受け取ることがあります。
10. クラークは情報を要求元のクライアント・アプリケーションに渡します。

前の例 (252ページの図25 および 253ページの図26) はいずれも、初期検索を表していることに注意してください。 CDS クラークは、GDA の位置を調べるとそれをキャッシュしておきます。クラークは調べた外部セル内のサーバー・アドレスもキャッシュしているので、それ以降に同じセル内の名前に関する要求があった場合は、直接外部サーバーにコンタクトすることができます。

また、GDA は自分のセル名を知っているので、自分のセル内の名前の検索にはグローバル・ディレクトリー・サービスにコンタクトせずに済むということにも注意してください。さらに、 GDA はセル名が X.500 または DNS のネーミング構文に合致しているかどうかを認識することができるので、その知識を利用して適切なグローバル・ディレクトリー・サービスに検索要求を経路指定します。 セル名が X.500 命名構文の規則に準拠している場合は、 GDA は最初に LDAP クライアントに要求を送り、 LDAP クライアント・サーバーでそれが解決されない場合は、それを GDS クライアントに送ります。

グローバル・ディレクトリー・エージェントの管理

DCE 構成プログラムを使用して GDA を構成します。 GDA は、構成後は、ほとんど管理を必要としません。 (GDA の構成の詳細については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド — 入門* を参照してください。)

GDA は、一般に、通常のシステム始動および遮断プロシージャーの一部として実行されるスクリプトによって、自動的に開始および停止されます。しかし、時には、コマンドを使用して GDA を停止し、再スタートしたい場合があります。 DCE 構成プログラムを使用して GDA を構成した後、以下のステップを使用して、GDA を開始したり停止したりすることができます。

GDA は **gdad** と呼ばれるプロセスとして実行されます。 **gdad** プロセスを開始するには、以下のステップに従います。

1. CDS サーバーがすでにセル内のどこかで実行されていることを確認します。
2. システムにスーパーユーザー (**root**) としてログインします。
3. 次のコマンドを入力して GDA およびすべての必要なデーモンを開始します。

```
# start.dce gda
```

GDA を停止するには、**stop.dce gda** を使用します。

他のセルがこのセルを検索できるようにする

GDA は、ローカル・セル内の CDS クラークが他のセルを検索できるようにするメカニズムです。自分のセルを他のセルからアクセスできるようにするには、現在サポートされているグローバル・ネーミング環境の 1 つの中にそのためのエントリーを作成しなければなりません。その前に、適切なネーミング機関から固有のセル名を取得します。 (詳細については *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド — 入門* を参照してください。)

セルを構成し、名前を付け、セル・ネームスペースを初期化した後、 **dcecp directory show** コマンドを使用して、 X.500、LDAP、または DNS 内のセル・エントリーを作成または変更するために必要なデータを入手できます。以下のサブセクションでは、 GDS、LDAP サーバー、または DNS サーバー内のセル・エントリーを定義し、保守する方法について説明します。これらのセクションは、X.500 と DNS に基本的に精通していることを前提としています。詳細については、各グローバル名サービスの該当する資料を参照してください。

ドメイン・ネーム・システムにおけるセルの定義

DNS 内の名前は、*リソース・レコード* と呼ばれる、1 つ以上のデータ構造と関連しています。リソース・レコードはデータ・ファイルに格納されており、その名前と位置は実装システムごとに異なります。セル・エントリーを作成する場合は、データ・ファイルを編集して、セル・ネームスペース・ルートのレプリカを保守している CDS サーバーごとに 2 つのリソース・レコードを作成しなければなりません。

最初のリソース・レコードは、タイプが AFSDB か MX であり、 CDS サーバーが常駐するシステムのホスト名を含んでいます。 MX を AFSDB の代わりに使用することができます。 2 番目のレコードは、タイプが TXT で、サーバーが保守するルート・ディレクトリーのレプリカに関する以下の情報を含んでいます。

- セル・ネームスペースの 16 進表記の UUID
- レプリカのタイプ (マスターまたは読み取り専用)
- レプリカが常駐するクリアリングハウスのグローバル CDS 名
- クリアリングハウスの 16 進表記の UUID
- クリアリングハウスが常駐するホストの DNS 名

次の例は、ルート・ディレクトリーの 2 つのレプリカが存在する、 **cs.tech.edu** という名前のセルの AFSDB リソース・レコードのセットを示しています。最初のリソース・レコードのみにセル名が含まれていることに注意してください。 2 番目、3 番目、および 4 番目のレコードは、セル名を含んでいないため、同一セルに関連するものと想定されます。 **TTL** という見出しは time-to-live (存続期間) の略語であり、これが過ぎると DNS キャッシュ内でデータが有効と見なされなくなる値 (秒単位) を示します。 (示されている値は、デフォルト値の 1 週間を指定しています。) **IN** クラスは、プロトコルがインターネットであることを示し、サブタイプの **2** は、レコードに示されているホスト上にネーム・サーバーが存在することを示します。

```
;First Replica:
;Name            TTL       Class     Type     Subtype   Host
cs.tech.edu      604800    IN        AFSDB     2        fox.cs.tech.edu
```

```
;Name            TTL       Class     Type    Rdata
                 604800    IN        TXT      (1        ;version
                 fd3328c4-2a4b-11ca-af85-09002b1c89bb   ;ns uuid
                 Master                                 ;Replica1 type
                 /.../cs.tech.edu/cs1_ch                ;ch name
                 fd3328c5-2a4b-11ca-af85-09002b1c89bb   ;ch uuid
                 fox.cs.tech.edu)                       ;host
;Second Replica:
                 604800    IN        AFSDB     2        rox.cs.tech.edu
                 604800    IN        TXT      (1        ;version
                 fd3328c4-2a4b-11ca-af85-09002b1c89bb   ;ns uuid
                 Read-only                              ;Replica2 type
                 /.../cs.tech.edu/cs2_ch                ;ch name
                 fd3429c4-2a4b-11ca-af87-09002b1c89bb   ;ch uuid
                 rox.cs.tech.edu)
;host
```

MX を AFSDB の代わりに使用することができます。次の例は、ルート・ディレクトリーの 2 つのレプリカが存在する、同じセル **cs.tech.edu** の MX リソース・レコードのセットを示しています。

```
;First Replica:
;Name            TTL       Class     Type    Preference   Exchange
cs.tech.edu.     604800    IN        MX      1            fox.cs.tech.edu.
;Name            TTL       Class     Type    Rdata
                 604800    IN        TXT     (1           ;version
                 fd3328c4-2a4b-11ca-af85-09002b1c89bb     ;ns uuid
                 Master                                   ;Replica1 type
                 /.../cs.tech.edu/cs1_ch                  ;ch name
                 fd3328c5-2a4b-11ca-af85-09002b1c89bb     ;ch uuid
                 fox.cs.tech.edu)                         ;host
;Second Replica:
                 604800    IN        MX      2
rox.cs.tech.edu.
                 604800    IN        TXT     (1           ;version
                 fd3328c4-2a4b-11ca-af85-09002b1c89bb     ;ns uuid
                 Read-only                                ;Replica2 type
                 /.../cs.tech.edu/cs2_ch                  ;ch name
                 fd3429c4-2a4b-11ca-af87-09002b1c89bb     ;ch uuid
                 rox.cs.tech.edu)
;host
```

セルを構成した後、**dcecp directory show** コマンドを使用して、前の例で示されたようなリソース・レコードを作成するために必要な情報を表示することができます。次に示すのは、 **directory show** コマンドを使用して **/.../cs.tech.edu** という名前のセルの出力を表示する例です。

```
dcecp> directory show /.../cs.tech.edu
```

DNS ネームスペース内に新しいリソース・レコードを作成するには、 **directory show** コマンドからの情報を使用して、適切にフォーマットされたデータを DNS データ・ファイルに入れます。

LDAP サーバーでのセルの定義

ldap_addcell ユーティリティーを使用すると、 DCE セル情報を LDAP サーバーから入手したり、動的に追加することができます。 **ldap_addcell** コマンドは、root 権限でもって実行する必要があります。 **ldap_addcell** コマンドは、次のことを行うことができます。

- セル・バインディングを使用して新しいディレクトリー・オブジェクトを作成する

- 既存のディレクトリー・オブジェクトを変更して、セル・バインディングを追加する
- 既存のディレクトリー・オブジェクト内のセル・バインディングの値を変更する
- 既存のディレクトリー・オブジェクトからセル・バインディングを削除する

ディレクトリー・オブジェクトに追加される、またはそこから検索されるセル・バインディングは、 X.500 サーバー (GDS) に使用されるのと同じ形式で、次の 2 つの属性で保管されます。

- CDSCELL
- CDSREPLICAS

ldap_addcell コマンドを使用すると、セルのための該当する情報を LDAP サーバーに加えることができます。セル・エントリー内のデータは、セル名の検索の後で GDA が CDS に渡すものです。一方、CDS はこの情報を使用してセル内のサーバーとコンタクトします。

userid や password などの認証情報は、 ldap_addcell ユーティリティーがディレクトリー・サービスに書き出すので、そのユーティリティー呼び出しの一部です。ディレクトリー・サービスに保管される DCE セル情報は、 X.500 登録ユーティリティーあるいは ldap_addcell 登録ユーティリティーのいずれで書き込んだ場合も、その内容は同じです。

ldap_addcell コマンドの構文は、次のとおりです。

```
ldap_addcell -h ldap_server -a authentication_DN -p password [-o object_class, ¥
object_class...]|[-d]
```

ここで、

-h *ldap_server*
: セル・バインディングを保持する宛先の LDAP サーバー名

-a *authentication_DN*
: 認証され、セル・バインディングの追加に使用される、 LDAP 名構文に指定されている識別名 (DN)

-p *password*
: 識別名 (DN) の認証に使用するパスワード

-o *object_class*
: 作成または変更するエントリー (登録) の属性 object_class の値 (複数も可)。複数の *object_class* 値をリストする場合は、コンマで区切る必要があります。

-d
: ディレクトリーの項目から DCE セル情報属性を削除します。ディレクトリー・エントリー全体の削除は行いません。

このコマンドは root 権限でもって実行しなければなりません。また、このコマンドはメッセージを stderr に印刷します。

次の **ldap_addcell** の例では、以下のことを想定しています。

- LDAP サーバー・マシン名は、bermuda.austin.ibm.com である。
- gdatest は、LDAP サーバーへの書き込みアクセス権を持っているユーザーである。

- gdatest は、ユーザー gdatest のパスワードでもある。
- organizationalUnit には、補助オブジェクト dceCellInfo を入れることができる。
- LDAP サーバーはスキーマ検査を行う。

例 1 は、LDAP での通常のセル・バインディング作成を示しています。

```
ldap_addcell -h bermuda.austin.ibm.com -a "cn=gdatest,ou=austin,o=ibm,c=us" -p "gdatest" ¥
       -o organizationalUnit,dceCellInfo
```

例 2 は、CDS_CELL および CDS_REPLICAS 属性の削除を示しています。

```
ldap_addcell -h bermuda.austin.ibm.com -a "cn=gdatest,ou=austin,o=ibm,c=us" -p "gdatest" -d
```

例 3 は、既存のオブジェクト内の CDS_CELL および CDS_REPLICAS 属性の変更を示しています。

```
ldap_addcell -h bermuda.austin.ibm.com -a "cn=gdatest,ou=austin,o=ibm,c=us" -p "gdatest
```

ldap_addcell コマンドのほとんどのパラメーターには対応する環境変数があり、**ldap_addcell** コマンドの呼び出し時にパラメーターが指定されていないと、この対応する環境変数が使用されます。 **ldap_addcell** のパラメーターと、その対応する環境変数は、次のとおりです。

ldap_addcell のパラメーター	**環境変数**
-h	LDAP_SERVER
-a	LDAP_AUTH_DN
-p	LDAP_AUTH_DN_PW
-o	LDAP_OBJECT_CLASS

注: -d パラメーターには対応する環境変数がありません。

セル・エントリーがすでに登録されている場合は、 CDS_CELL および CDS_REPLICAS の属性は、 **-d** パラメーターが指定されていない限り、このセルのための新しい値でもって置き換えられます。

CDSCELL および CDSREPLICAS で置き換えられる CDS_CELL および CDS_REPLICAS

LDAP プロトコルのバージョン 3 は、下線 (_) を CDS_CELL および CDS_REPLICAS の正しい文字としてサポートしていないので、それぞれ CDSCELL および CDSREPLICAS に変更されています。 LDAP サーバー内に X.500 セル名を登録するために DCE 2.2 の初期バージョンを使用した場合、 **gdad** サーバーを実行する各システム上に DCE 2.2 の最新バージョンをインストールしなければなりません。次に、**ldap_addcell** コマンドを使用して、 DCE セル情報を再登録します。**ldap_addcell** コマンドは、 DCE 2.2 の最新バージョンにアップグレードされているシステム上で実行しなければなりません。 LDAP サーバー内のスキーマは、CDS_CELL および CDS_REPLICAS の代わりにそれぞれ CDSCELL および CDSREPLICAS を使用するようにも更新されなければなりません。

GDA がサポートしていない SLAPD 照会

Standalone LDAP Daemon (SLAPD) がサポートしている照会には、現在以下の制限事項があり、外部セル・バインディングを解決する際には GDA がそれらの照会を使用できないようになっています。

ldap_search_s( ) API が使用する SLAPD 照会では、検索の有効範囲が LDAP_SCOPE_SUBTREE でなければなりません。つまり、照会ポイントの下にあるサブツリー全体が検索されなければなりません。 GDA は、必要な固有のオブジェクトを認識しており、 LDAP_SCOPE_BASE の有効範囲で検索を行います。 GDA は、**ldap_search_s( )** の結果から複数のオブジェクトを受け取ることはできません。

第5部 DCE 分散タイム・サービス

第23章 DCE 分散タイム・サービスの紹介

DTS はソフトウェア・ベースのサービスであり、ローカル・エリア・ネットワーク (LAN) および広域ネットワーク (WAN) 内のシステムに対して耐障害性のある正確なクロック同期を提供します。 DTS の提供するクロック同期により、分散コンピューティング・アプリケーションはイベントの順序付け、期間、およびスケジューリングを判別することが可能になります。

DTS は、一連の連携するシステム上のソフトウェア・コンポーネントからなり、DCE で使用されるクライアント / サーバー・モデルに適合しています。 DTS 実装システムでは、各サーバーはクラーク と呼ばれる仲介を通じて、多数のクライアント・システムおよびアプリケーションに時刻を提供します。クラークは、クライアント・システムに常駐しています。 (当ガイドのこの部分では、エンティティー という用語は、サーバー・プロセスとクラーク・プロセスが同じ機能を持つ場合は、サーバー・プロセスまたはクラーク・プロセスを指しています。)

大部分の DCE ノードには DTS クラークがあり、これがそのクライアント・システム上のクロックを調整しています。クラークは、リモート・プロシージャー・コール (RPC) を使用して、ネットワーク内の 1 つ以上のサーバーから時刻値を取得します。 DTS クラークを持たないノードには DTS サーバーがあります。サーバーは、クラークに時刻値を提供する他に、自身のホスト・システムのシステム・クロックの調整も行います。サーバーはまた、ネットワークの外部の標準時刻のソースから基準時刻値を取得することもできます。

どのデバイスもある特定の瞬間の正確な時刻を測定することができないので、 DTS は時刻を、正しい時刻を含んだ、インターバル (間隔) として表示します。 DTS モデルでは、クラークは複数のサーバーから時間間隔を取得し、それらがオーバーラップする交点を計算します。その後でクラークはそのクライアント・システムのシステム・クロックを調整して、計算した交点の中央に合わせます。クラークは、大多数の時間間隔と交差しない時間間隔を受信すると、その交差しない値をフォールトと宣言します。クラークは、新しい時刻を計算するときにはフォールト値を無視することによって、欠陥のあるサーバー・クロックがクライアントに影響を与えないようにします。

DTS では、米国国立標準・技術研究所 (NIST) などの外部ソースからの時刻値のインポートも許可します。 DTS は、グリニッジ標準時 (GMT) に代わって広く用いられている、協定世界時 (UTC) を基準として使用します。多くの標準組織は無線、電話、および衛星によって UTC を広めており、これらの信号を受信して解釈する、市販のデバイス (タイム・プロバイダー) が入手可能です。 DTS は、タイム・プロバイダー・プロセスが DTS サーバーに UTC 時刻値を渡し、その値をネットワーク内に伝搬する方法を記述した、タイム・プロバイダー・インターフェース (TPI) を提供しています。 TPI では、他の分散タイム・サービスが DTS と相互に作用することも許します。

DTS は、分散アプリケーションを実行するコンピューター・ネットワークに対して、他にも多数の有用なサービスを提供しています。 DTS の主な機能と利点を以下に示します。

- 正確性 - DTS はクライアントが正確な時刻を受信する可能性を最大にします。DTS は基本的基準として協定世界時 (UTC) を使用し、 UTC を含んだ時間間隔はいずれも正しいと定義します。
- 定量的時刻測定 - DTS は特定の測定基準とメーカーの仕様を用いて、サーバーが報告した時刻の品質を判定します。
- 耐障害性 - DTS は障害のあるサーバーを報告し、クロックの同期にはその値を使用しません。
- 管理能力 - DCE 制御プログラム (**dcecp**) は、ソフトウェアを制御し、モニターできるようにします。
- アプリケーション・プログラミング・インターフェース (API) - DTS は、アプリケーションが UTC 時刻値の取得、比較、および計算を行えるようにする、インターフェースを提供します。
- ローカル時刻変換 - 時刻値を表示する際に、 DTS は内部で使用している UTC 時刻をローカル時刻値に変換します。
- 単調性 - DTS は通常は単一方向のクロック調整を行います。しかし、非単調クロック調整に DCE 制御プログラムを使用することもできます。
- 自動構成 - DTS エンティティーは、 RPC プロファイル (検索テーブル) を使用して、ローカル・エリアまたはセルの中のサーバーの位置を取得します。
- 効率 - 複雑な部分はサーバーの中に置かれ、ネットワーク・オーバーヘッドは最小になっています。

DTS の利点

DTS は、タイム・サービスが通常提供する機能をすべて提供しますが、ネットワークのパフォーマンスを向上させる機能もいくつか持っています。次のサブセクションでこれらの DTS の機能について説明します。

- アプリケーションのサポート
- 外部タイム・プロバイダーのサポート
- 管理可能性
- 誤差の定量的測定

アプリケーションのサポート

オペレーティング・システムと分散アプリケーションには、プロセスを調整するための、同期化された時刻測定値が必要です。 DTS はネットワーク内のシステム・クロックを互いに同期化し、外部タイム・プロバイダーが存在する場合には、UTC 時刻標準に同期化します。システム・クロックを読み取る分散アプリケーション (大部分のアプリケーションがこれに当たる) には DTS が必要です。ネットワーク内の分散アプリケーションとシステムの数が増加するにつれて、プロセス調整のために DTS はますます重要になります。

DTS がシステム・クロックに提供する、同期化された時刻を使用する既存のアプリケーションにはいくつかのタイプがあります。これらのアプリケーションは、ネットワーク全体で起こるイベントを調整するために、同期化されたシステム・クロックを参照しなければなりません。アプリケーションは以下に示す機能のために、同期化されたクロックを使用します。

- イベントの測定 - アプリケーションは、システム・クロックを読み取ってタイマーを開始したり停止したりすることができ、また、イベント間の経過時間を測定することができます。
- イベントの報告 - アプリケーションは、イベント発生時にクロックを読み取り、イベント報告書にタイム・スタンプを付加することができます。
- イベントのスケジューリング - アプリケーションは、システム・クロックを読み取り、相対時間を加えて、将来のイベントの発生を決定することができます。
- イベントの順序付け - アプリケーションは、同期化されたシステム・クロックから得られたイベント・レポート・タイム・スタンプを読み取って、イベントの順序を判別することができます。

新しいアプリケーションには、DTS が API を提供します。この API は、新しいアプリケーションが 2 進タイム・スタンプを取得して処理するのに使用できる、ルーチンを提供します。 DTS API は ANSI C 言語構造体をサポートしています。 (DTS API の詳細については、 *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* を参照してください。)

外部タイム・プロバイダーのサポート

大部分のネットワークの場合、システム・クロックを UTC の時刻標準と同期化させるのが理想的です。多くの市販のデバイスを利用して、標準団体が提供する UTC 時刻を取得することができます。このようなデバイスは、短波ラジオや、衛星、電話によって信号を受信します。使用しているネットワークまたはセルが単一の LAN よりも大きい場合は、少なくとも 1 つの外部タイム・プロバイダーを DTS ソフトウェアと組み合わせて使用されることをお勧めします。 (タイム・プロバイダー・ハードウェアの提供会社のリストについては、 589ページの『付録C. タイム・プロバイダーとタイム・サービス』を参照してください。タイム・プロバイダーのプログラムのサンプルが *dcelocal*/**usr/examples/dts** にあり、オンラインで利用できます。)

DTS サーバーは、TPI を使用してタイム・プロバイダーと同期化することができ、これについては *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* に記述されています。 TPI は、 DTS サーバー・プロセスとタイム・プロバイダー・プロセスとの間の通信を指定します。

DTS サーバーは、同期の際に TPI を使用して、タイム・プロバイダー・プロセスをチェックします。タイム・プロバイダーが存在する場合、サーバーはそのタイム・プロバイダーだけと同期化します。タイム・プロバイダーが存在しない場合、サーバーはネットワーク内の他のサーバーと同期化します。

タイム・プロバイダーを DTS サーバーとともに使用すれば、サーバーを UTC と緊密に同期化させることができます。他のサーバーが、タイム・プロバイダーを持つサーバー (TP サーバー) に対して時刻を要求すると、その TP サーバーの正確な時刻がネットワーク全体に伝搬されます。 (タイム・プロバイダーとサーバー同期化プロセスに関する詳細については、 266ページの『DTS の基本概念』を参照してください。)

管理可能性

DTS 同期機能は背景プロセスとして実行されます。 DTS が最初に構成されて以後は、システム・クロックの同期をとるためのシステム管理者からの入力はほとんど、またはまったく必要ありません。さらに、DTS には耐障害性があります。 DTS は、誤動作しているクロックが誤った時刻をネットワーク内の他のクロックに提供するのを防止します。ただし、時にはシステム管理者が以下の機能を実行しなければならない場合があります。

- システム・クロック問題の識別
- システム・クロックの調整
- ネットワーク条件の変化による DTS 属性の変更
- ネットワーク・トポロジーが変化したときのシステム構成の変更

DTS は、システム・クロックの調整、DTS 管理パラメーターの値の変更、およびネットワークのサーバーの追加または除去をシステム管理者が行うことを可能にする、全機能を備えた管理インターフェースを提供します。

システム・クロックに関する問題の解決を支援するために、DTS は、システム・クロックが不正確になったり同期に失敗したりするなどのまれな事態の発生時に、システム・オペレーターと管理者に通知するイベント報告機能を備えています。

誤差の定量的測定

他のネットワーク・タイム・サービスと異なり、 DTS はメーカー仕様と実観測を使用して、 UTC と比較したシステム・クロックの誤差を判別します。 DTS は、自身が内部で使用する、各時刻値に誤差の測定値を付加します。この測定値は、累積クロック・エラー、通信遅延、および処理遅延を考慮します。 DTS は、1 つまたは複数のソースからの時刻と誤差の測定値を結合して、クライアント・システムに対する最も正確な新しいクロック設定値を計算します。 (DTS 同期化プロセスに関する詳細については、 268ページの『システム・クロックの同期』を参照してください。)

DTS の基本概念

このサブセクションでは、システム・クロックとネットワークの特性、DTS の同期の概念、 DTS のクロック調整、および DTS の時刻表現について説明します。システム管理者は、これらのサブセクションを読んで DTS の基本概念を理解してから、285ページの『第25章 DCE DTS の管理』に進んでください。

時刻測定の要因

次のサブセクションでは、時刻測定に影響する要因について説明するとともに、DTS がそれらを処理する方法について説明します。

クロック・エラー

すべてのシステム・クロックは、クロック・エラーの一因となり、同期化処理を妨害する、共通の特性を持っています。システム・クロック・エラーは、時間の経過とともに増加する傾向があります。エラーの変化率は*ドリフト* と呼ばれます。ネットワーク内の各システム・クロックが同時に開始され、同じ速度で実行されている

場合、クロックは同期がとれているでしょう。しかし各システム・クロックが異なる率でドリフトするために、ネットワーク全体のシステム・クロックの同期が取れなくなります。

2 つのクロックの読み取り値の差のことをクロック間のスキュー ともいいます。多数のコンピューター・システムで使用されているクロックは、 1 日当たり数秒という最大ドリフトが指定されています。数日間修正しないでおくと、ネットワーク内のシステム・クロック間のスキューが分散アプリケーションの性能を抑制する可能性があります。

各ノード上の DTS サーバーまたはクラークは、そのクライアントのシステム・クロックのドリフトを追跡し、定期的に他の DTS ノードと同期を取り、そのクライアントの時刻値と他の DTS ノードの時刻値との間のスキューが小さくなるようにしています。 DTS サーバーまたはクラークは、この繰り返し行われる同期化処理の最終ステップとして、そのクライアント・ノード上のシステム・クロックの調整を行います。

通信と処理の不確実性

通信遅延もまた、特に 2 つのシステムが WAN または低速リンクを使用して通信する場合に、同期化処理を抑制します。 DTS は、システム間のメッセージの送受信に必要な既知の処理遅延に合わせて調整することができますが、通信リンクの品質がさまざまなため、メッセージの送受信および確認に要する時間はメッセージによって異なります。このような遅延は、ネットワークの通過時間と、着信タイム・スタンプの読み取りに要する時間の両方が変化するため、正確に知ることはできません。

通信遅延と処理遅延の見積もりを使用する代わりに、 DTS は、ネットワークを介して送信された時刻測定値を伴う、すべての既知のエラー要因を記録します。この測定値によって、DTS は、地理的位置や通信リンク上の条件の変化とは関係なく、タイム・ソースの相対的品質を判断することができます。

誤差値

システム・クロックを最も正確な設定値に同期させるために、 DTS は、互いに比較した、および UTC と比較した時刻ソースの正確さを判断する方法を必要とします。このセクションは、ネットワーク内で使用可能な、時刻ソースの相対的な正確さを DTS が判断する方法について説明します。

DTS は、誤差値 (または許容度) を使用して、システム・クロックと外部タイム・プロバイダーから得られる、時刻値の相対的な正確さを判別します。この DTS の機能は、それぞれの時刻値を、連続体の 1 点としてではなく、インターバル、すなわち、範囲として効果的に変形します。

誤差値は、以下の 3 つの要因によって決まります。

- ドリフト - クロックを読み取るとき、DTS は、前に DTS がクロックを読み取った後にクロックがドリフトした可能性のある時間量を計算します。大部分の誤差値の最大のコンポーネントがドリフトです。
- 通信遅延 - 誤差には、システム間の通信遅延の不確実部分も含まれます。 DTS は処理遅延を補正しますが、ネットワーク・リンクで起こる遅延の変化を予測したり、直接測定したりすることはできません。クラークまたはサーバーが同じ

LAN 上のシステムから取得する誤差値は、その LAN の外側のサーバーから取得する値よりかなり低くなる傾向があります。

- うるう秒 - UTC 時刻は非常に正確な原子時計によって測定されています。しかし、標準では地球の位置に基づいて時刻を保持しています。地球の自転の遅れのために、ときどき UTC 時刻を 1 秒だけ進める必要があります。このイベントはうるう秒 と呼ばれます。うるう秒は、いずれかの月の最終秒に発生する可能性があり、通常は約 18 カ月に 1 回発生します。それぞれの月の終わりに、 DTS はすべての誤差測定値を 1 秒だけ増加させて、うるう秒に対応します。 DTS は、うるう秒が起こらなかったと外部タイム・プロバイダーが判断した場合、後でクロックを調整し、誤差の余分の秒を除去します。

DTS が訂正しなければ、システム・クロックの誤差は常に増加していきます。たとえば、クロックが UTC 時刻の 0:00:00.00 (真夜中) に開始し、誤差はゼロであるとします。ドリフトのために、クロックが次に 0:00:00.00 の時刻を示したとき、誤差は 8 秒です。 UTC 時刻は 23:59:52.00 または 0:00:08.00 の可能性がありますが、おそらくこの間のいずれかの値でしょう。したがって、 図27 に示されるように、システム時刻は、UTC 時刻を含み、しかも誤差値で囲まれたインターバルです。時刻の表示に DTS 形式を使用すると、時刻と誤差インターバルが結合された値は以下のように表されます。

```
1993-08-03-00:00:00.000I08.000.
```

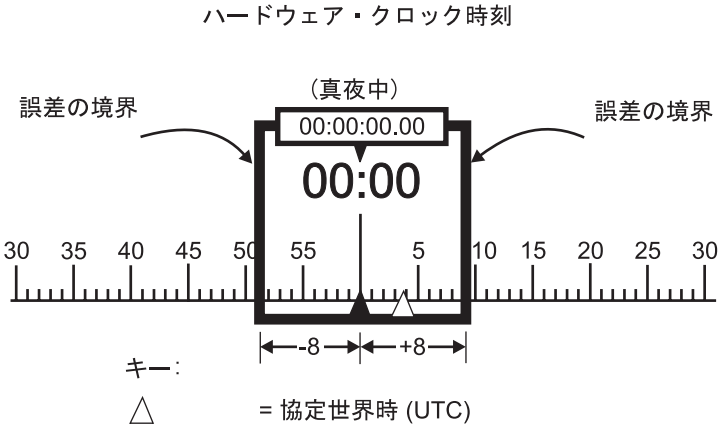


図 27. 時刻と誤差

システム・クロックの同期

一様なシステム時刻を維持するために、 DTS のサーバーとクラークは定期的にネットワーク・システム内のすべてのクロックを同期させます。各システム上の DTS エンティティーは、サーバーがクロックと誤差値の組み合わせ (時間間隔) を要求元のシステムに送信するように要求することによって、この同期を実行します。次にエンティティーは、サーバーから送られた値を使用して、新しいシステム時刻を計算します。

DTS のサーバーとクラークとでは、同期化手順が多少異なります。他のシステムとの同期を試みる前に、 DTS サーバーは外部タイム・プロバイダーがサーバー・シ

ステムに存在するかどうか必ずチェックします。タイム・プロバイダーがない場合は、そのサーバーは他のサーバーに時刻を要求します。使用可能なタイム・プロバイダーがなく、サーバーが自身のピア・サーバーと同期する場合は、サーバーは新しいシステム時刻の計算の際に、そのサーバー自身のシステム時刻を入力値の 1 つとして使用します。

大部分のネットワーク・システムは DTS クラーク・プロセスを実行します。クラークはタイム・プロバイダーを持つことができず、新しい時刻の計算にクラークのクライアント・システムのシステム時刻を使用しません。クラークは、クライアント・システムのクロックと同期をとろうとするとき、サーバーから得た時刻値だけを使用して、新しいシステム時刻を計算します。

DTS クラークは、複数のサーバーに時間間隔を要求する場合、それらを使用して、正確で (つまり、UTC を含んだ)、 かつ誤差が最小の新しい時刻を計算します。サーバーが応答し、 DTS クラークがそれぞれの時刻値のネットワーク通信の不確実性とドリフトを計算すると、クラークは 1 セットのインターバルを得ます (270ページの図28 の **t1** から **t4** まで)。それぞれのインターバルには UTC が含まれているので、その交点はクラークが選択することができ、かつ UTC を含んだ最小のインターバルです。この交点が計算時刻です。 DTS エンティティーは、計算時刻を使用して、サーバー値を受信するシステムのクロックを調整します。

同期の際に大きな誤差値を除去する他に、DTS はフォールト・クロックから受信したインターバルも廃棄します (図の **t2**)。 DTS は、大部分のインターバルと交差しないクロック・インターバルを検出して、それを拒否します。 DTS がフォールト・インターバルを検出すると、そのフォールト値を送信したサーバーを識別するイベント・メッセージを表示して、システム管理者に通知します。

高ドリフトのクロックを持つサーバーやネットワーク内で遠方にあるサーバーは、その時刻を DTS エンティティーに渡しますが (図の **t1**)、時間間隔が大きいと、それより正確な時刻を利用できるので、その時間間隔は無視されます。 270ページの図28 では、正確な時刻 (**t1**) の終了点は、フォールトと宣言されたインターバル (**t2**) の終了点よりも計算時刻の中心点から離れていることに注意してください。

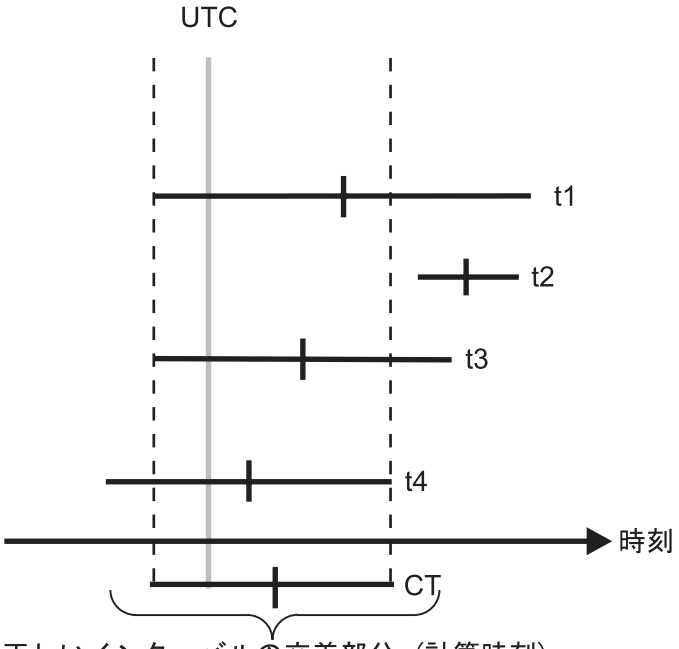


図 28. 計算時刻

同期化処理の際には、最大の正確度を持つサーバーが、ネットワーク全体の新しいシステム時刻の決定に最も影響します。前記の図では、時刻値 **t3** を出したサーバーの正しいインターバルが最も小さいので、計算時刻に最も近いことになります。通常は、外部タイム・プロバイダーを持つサーバー・システムが最も正確な時刻を持つサーバーです。 TP サーバー以外では、最も高品質のクロックと最良の通信リンクを持つサーバーが、他のシステムの時刻に影響を与える度合いが最も高いという傾向があります。

同期化処理によっても、システム間のスキューが削減します。計算時刻インターバルは、どのクロックが提供するインターバルよりも小さくなる場合がよくあります。前記の図の計算時刻インターバルは、どのソース・インターバルよりも小さいことに注意してください。同期化手順がそれぞれのネットワーク・システムで定期的に繰り返されるにつれて、システム間のスキューが減少し、システムはより緊密に同期化されます。ただし、ネットワークにタイム・プロバイダーが存在しない場合は、クロックがまとまって UTC からドリフトする可能性があります。

DTS によるシステム・クロックの調整方法

多くのシステム・クロックは発振器に基づいており、ハードウェアとソフトウェアの組み合わせによって動作します。それぞれのクロックのハードウェアには、固定したインターバルでオペレーティング・システムに対して割り込みを送るタイマーがあり、それぞれの割り込みが 1 つのティックです。時刻の現行値を含んでいるソフトウェア・レジスターは、それぞれのティック時に一定量 (例: 10 ミリ秒) だけ増やされます。 DTS は、ソフトウェア・レジスターに加算される増分値のみを変更することによってクロックの速度を調整し、ハードウェア・クロックのチックには直接影響を与えません。

DTS は 100 対 1 の割合でシステム・クロックを調整します。したがって、時間単位 1 のエラーの調整には時間単位 100 が必要です。たとえば、 1 秒のエラーの訂正には 1 分 40 秒かかります。この調整率は通常のドリフト率を超えているので、クロックからこれ以上介入されずに、同期が行われます。

図29 は、 DTS がソフトウェア・レジスターの増分を変更する方法を示しています。一番上の線が 10 ミリ秒のチックごとに正常なクロックに 10 ミリ秒の増分を加えることを表しています。中央の線は速いクロックの調整を示しており、 DTS がティックごとに 10 ミリ秒ではなく、9.9 ミリ秒だけレジスターを増加させて、クロックを遅くします。一番下の線は遅いクロックの調整を示しており、 DTS がティックごとに通常の 10 ミリ秒ではなく、 10.1 ミリ秒だけレジスターを増加させて、クロックを速めます。

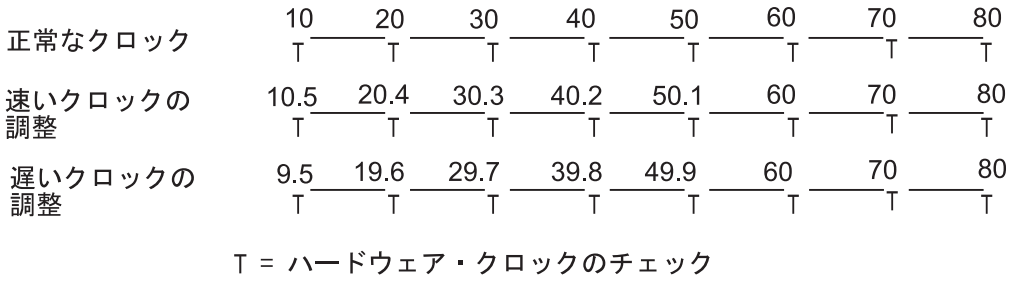


図 29. クロックの調整

時には、システム・クロックを徐々に調整するよりも、直ちにセットした方がよい場合があります。 DTS には、次の場合に備えて、このオプションが用意されています。

- システム始動時に初期システム時刻を設定したいとき
- 最後に同期をとって以来長時間が経過し、システム・クロック間のスキューが大きくなりすぎたために、それが徐々に調整されるのを待っていることはできないと判断した場合
- ネットワークに重大なハードウェア障害があり、そのために多数のクロックがフォールトになっているとき
- あるクロックの時間間隔が他のクロックの時間間隔と交差せず、エラーがあらかじめ定めた許容度を超えたとき

DTS の時刻表現

協定世界時 (UTC) は、グリニッジ標準時 (GMT) に代わって広く用いられている、国際的時刻標準です。この標準は国際標準時局 (BIH) によって管理されており、広く用いられています。内部プロセスのすべてに対し、 DTS は UTC を表すオペイク 2 進タイム・スタンプを使用します。 DTS の 2 進タイム・スタンプを読み取ったり、逆アセンブルしたりすることはできません。 DTS API を使用すれば、他のアプリケーションはタイム・スタンプを変換したり操作したりすることができますが、表示はできません。 DTS はまた、クライアント・システムで表示するために、 2 進のタイム・スタンプを ASCII テキストに変換します。

絶対時間

絶対時間は、時間目盛り上の 1 つの点です。 DTS の場合、絶対時間とは UTC の時間目盛りを指します。絶対時間の測定値はシステム・クロックまたは外部タイム・プロバイダーから得られます。 DTS がシステム・クロック時刻を読み取ると、その時刻はオペイク 2 進タイム・スタンプに記録されますが、そこには誤差やその他の情報も記録されています。 DCE 制御プログラム (**dcecp**) の **clock show** コマンドを使用して絶対時間を表示すると、その値は次の表示のような ASCII テキストに変換されます。

```
1993-11-21-13:30:25.78523-04:00I010.0825
```

DTS はすべての時刻を ISO 準拠フォーマットで表示します。上記の表示例を生成した国際標準化機構 (ISO) フォーマットの詳細を図30 に示します。

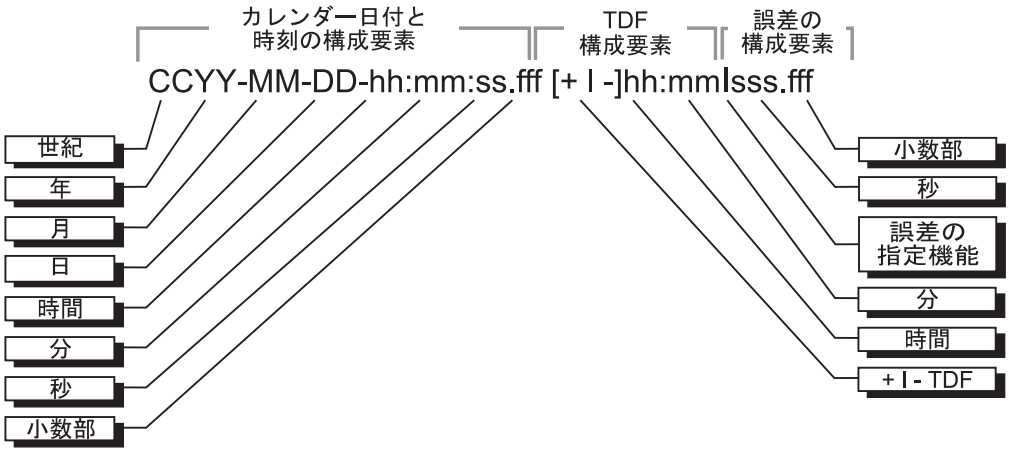


図 30. ISO 準拠時刻フォーマット

上の図で示されたフォーマット例で、相対時刻の前に **+** (正符号) または **-** (負符号) が付いているのは、カレンダーの日付と誤差が UTC からずれている時間と分を示しています。これらの文字のいずれかがストリングにあると、カレンダーの日付と時刻がシステムのローカル時刻であり、 UTC ではないことも示します。区切り文字 I は、時刻と関連した誤差のコンポーネントの始まりを示します。表示する DTS 時刻の表現方法は数通りあります。 BNF フォーマットの DTS 時刻については、593ページの『付録D. DTS 拡張 BNF』に定義されています。

dcecp clock show コマンドはすべての時刻を前記のフォーマット (図30 参照) で表示しますが、インターフェースは、入力に ISO フォーマットの次のようなバリエーションがあっても受け入れます (273ページの図31 を参照)。

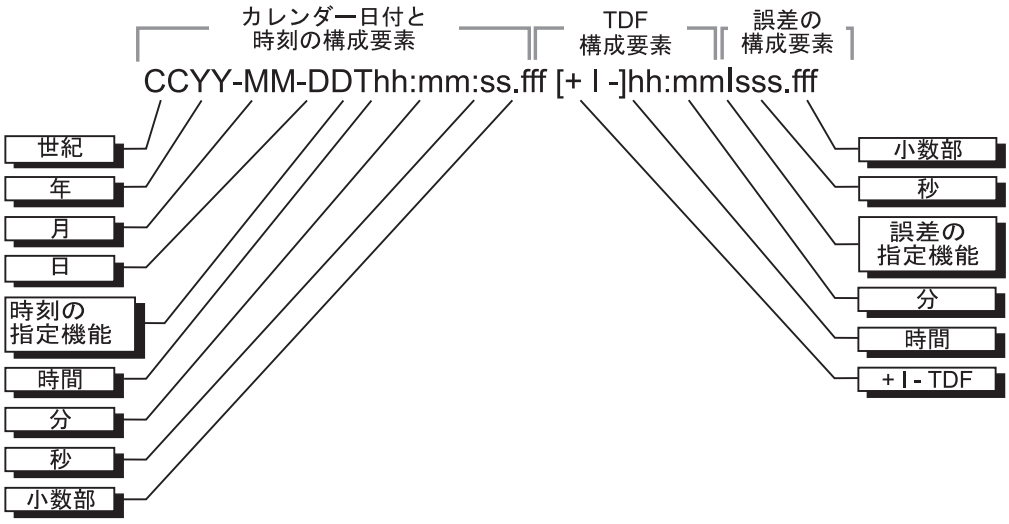


図 31. ISO 準拠時刻フォーマットのバリエーション

前の例では、輪郭線 T はカレンダーの日付を時刻と分離し、 , (コンマ) は秒を小数部と分離し、 ± (正符号または負符号) は誤差コンポーネントの始まりを示しています。

DTS は、時刻が表示されるときにはいつでも、 UTC ベースの絶対時間をローカル時刻に変更する変換機能を提供します。表示されるローカル時刻は UTC に時差 (TDF) を加えて得られます。なお、TDF は正または負の値をとり得ます。前記の例で、ストリング [**+/-**] *hh*:*mm* は TDF を示しています。システムのインストール時にシステムの時間帯規則を選択しますが、これによって、TDF および、TDF に対する季節的変更が決まります。初期起動の後、それ以後のすべての出力時刻はローカル時刻を示します。システムによって絶対時間が表示され、それに TDF 情報が含まれていなければ、それは UTC 時刻です。

相対時間

相対時間は、通常、別の時刻に対して加算または減算される、離散的な時間間隔です。絶対時間と関連した TDF が相対時間の一例です。相対時間は、通常、コマンドまたはシステム・ルーチンへの入力として使用されます。

図32 に、相対時間のフォーマットを示します。

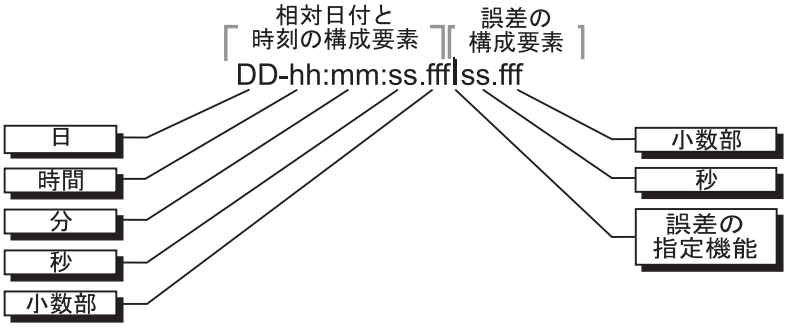


図 32. 相対時間のフォーマット

DTS 関連の **dcecp** コマンドで指定する単純な相対時間は、カレンダー日付フィールドも誤差フィールドも使用しません。それは、これらのフィールドが絶対時間と関連しているためです。正の相対時間には符号を付けませんが、負の相対時間の前には **-** (負符号) を付けます。

次の例は一般的な DTS 関連の **dcecp** コマンドで使用される相対時間を示しています。

```
21-08:30:25.000
```

単純相対時間はしばしば他の相対時間または絶対時間から減算したり、それに加算したりします。たとえば、「1 時間後に会おう」という場合は、現在の絶対時間に相対時間 +01:00 を加算します。相対時間と絶対時間を加算または減算するとき、入力される絶対時間の誤差が結果の絶対時間に持ち越されることに注意してください。たとえば、 1993-11-30-00:30:25.000I00.030 マイナス 00-00:15:25.000 は 1993-11-30-00:15:00.000I00.030 になります。

DTS の働き

DTS には次の 2 つの主なソフトウェア・コンポーネントがあります。

- クラーク
- サーバー

以下のサブセクションでは、これらのコンポーネントのそれぞれについて説明し、これらがクライアント・アプリケーションに時刻を提供するため、およびシステム・クロックを同期させるために、どのように関連するかについて説明します。

クラーク

DTS サーバーでないシステムはいずれも DTS クラークです。大部分のネットワーク・システムはクラーク・ソフトウェアを実行します。クラークはサーバー・リストを保守し、DTS クライアント・システムのために同期機能を実行します。

サーバー・リストを作成し、そのリスト上のサーバーと同期をとるためには、クラークはサーバーを自動的に見付けることができなければなりません。クラークは、リモート・プロシージャー・コール (RPC) プロファイルを使用して、サーバーを見付けます。プロファイルは、次のタイプのエントリーを含んでいる探索テーブルであることを思い出してください。

- サーバー・エントリー - 個々のリソース・プロバイダーの CDS 名。
- サービス・グループ・エントリー - 1 つの CDS 名で識別される、リソース・プロバイダーのグループ。
- プロファイル・エントリー - 他の構成プロファイルの名前。これらのエントリーは階層的にネストしたプロファイルを許します。

それぞれの DTS クラーク・ノードは最高 3 つのプロファイルを含みます。サーバーを見つける際に、クラークはまず、*ノード初期プロファイル* と呼ばれる、基本プロファイル内のエントリーの RPC 検索を実行します。次に、クラークは LAN プロファイル・エントリーを検索します。 LAN プロファイル・エントリーが検出されなかった場合は、クラークはデフォルト・プロファイル・エントリーを探します (デフォルト・プロファイルに LAN プロファイル・エントリーが含まれていること

があります)。クラークは LAN プロファイルを見付けると、サーバー・エントリーを読み取って、ローカル・サーバーのリストを作成します。この処理は、セット・インターバルごとに繰り返されます。

クラークは、 DTS 管理属性の **minservers** によって指定されただけの十分なサーバー・エントリーを取得していない場合、通常 LAN の外部にある、追加のサーバーを見つけようとします。これらのサーバーを見付けるために、クラークはウェルノウン CDS 名を持つセル・プロファイルを探します。セル・プロファイルには、グローバル・サーバー・エントリー (すなわち、通常は LAN の外側にあるサーバー) が含まれています。 (サーバーに関する詳細については、『サーバー』を参照してください。)

十分なエントリーが入ったサーバー・リストが構築されると、クラークはそのリスト上の複数のサーバーに対して時刻値を直接要求できるようになります。その後、クラークはこれらの時刻値を受信し、それらを使用して、クライアント・システムのための新しいシステム時刻を計算します。

サーバー

サーバーは、DTS のために多くの通信および同期の機能を提供します。クラークと同様、サーバーも他のサーバーに関する情報を LAN とセル・プロファイルからインポートします。ただし、サーバーは自身の CDS ネームスペース・エントリーへのバインディングのエクスポート、および LAN やセル・プロファイルへの自身の名前のエクスポートも行います。 (サーバーの構成および検索の方法の詳細については、以下のサーバー・サブタイプに関するサブセクションを参照してください。)

外部タイム・プロバイダーはサーバーに接続することができ、サーバーはタイム・プロバイダーから取得した正確な時間間隔をネットワーク全体に伝搬します。

サーバーが別のサーバーから時刻値を獲得するためには、両サーバーが同じエポック番号を持っていなければなりません。エポックは、DTS 実装システムを論理的に分離された区域に分割します。サーバーは、同じエポック番号を持つ他のサーバーだけと同期します。すべてのサーバーは、作成時には同じエポック番号を持っています。まれに、問題を訂正するためにネットワークからサーバーを分離するべく、管理インターフェースを使用して、サーバーのエポック番号を変更したい場合があります。

ローカル・サーバー・セット

ローカル・サーバーは、同じ LAN 上に常駐し、互いに同期することによって、クロックを保守しています。このタイプのネットワークはスループットが高いので、LAN 上のローカル・サーバー間のスキューは通常、200 ミリ秒未満に維持されています。ローカル・セット内のサーバーの中で少なくとも 1 つのサーバーが正確なタイム・プロバイダーと同期していれば、各サーバーにおける誤差はより小さくなる可能性があります。

サーバーは、最初に初期設定されるときに、そのバインディングをネームスペース内のそのエントリーにエクスポートし、その名前のエントリーを LAN プロファイルに追加します。すべてのサーバーは、ネットワークの関連部分の LAN プロファ

イルに自動的に入力されます。ローカル・サーバーも、同期できるサーバーのリストを作成するために、 LAN プロファイルからバインディングをインポートします。

ローカル・サーバーは、時間間隔を計算し、クロックを調整し、さらに同期化のためにお互いに時刻値を提供します。各サーバーは、一定のインターバルで、ローカル・セット内の他のサーバーすべてとの同期化を試みます。これより長いインターバルで、クラークは、ローカル・サーバーに対して時刻値を要求します。ただし、クラークは、 **minservers** 属性によって決定される数のサーバー (通常は、すべてのローカル・サーバーのサブセット) にだけインターバルを要求すれば十分です。

グローバル・サーバー・セット

ローカル・サーバーを使用できるのは、1 つの LAN 内のサーバーとクラークだけですが、グローバル・サーバーはセル全体で使用できます。どのサーバーもローカル・サーバーまたはグローバル・サーバーのいずれかとして構成することができます (DCE 制御プログラムの **dts configure** コマンドを参照)。グローバル・サーバーの数は通常は少ないのですが、グローバル・サーバーには、 DTS によるネットワーク内のすべてのノードの同期を可能にする、いくつかの重要な機能があります。グローバル・サーバーは次のような場合に必要です。

- ネットワークに複数の LAN または 1 つの拡張 LAN がある場合
- LAN 上にないシステムが、2 地点間リンクを介して LAN にアクセスするとき
- クラークまたはローカル・サーバーが、 **minservers** 属性により決められた数のローカル・サーバーをアクセスできない場合

-global オプションを指定した **dcecp dts configure** コマンドを使用することにより、ローカル・サーバーをグローバル・サーバーとして再構成することができます。サーバーをグローバル・サーバーとして構成すると、そのサーバーはそのバインディングをネームスペース内のそのエントリーに、そしてその名前をセル・プロファイルに、それぞれエクスポートします。

ローカル・サーバーおよびクラークは、 **minservers** 属性により必須と指定された数のローカル・サーバー応答を取得できないと、グローバル・サーバーに対して時刻値を要求します。一部のローカル・サーバーも、定期的にグローバル・サーバーに対して時刻を要求します。

クーリエ

クーリエ と呼ばれるローカル・サーバーは、同期が行われるたびに、ランダムに選択された 1 つのグローバル・サーバーに対して時刻値を要求します。 DTS の始動時に、サーバーの **courierrole** 属性値は自動的に **backup** に設定されます。サーバーのクーリエの役割は、この属性値を手作業で変更することによって、変更することができます。このためには、 **-change** オプションを指定した **dcecp dts modify** コマンドを使用します。サーバーが外部タイム・プロバイダーと接続されている場合、そのサーバーをクーリエとして再構成したい場合があります。

クーリエは、セル・プロファイルからバインディングをインポートしたグローバル・サーバーのリストを維持しています。どの同期化においても、クーリエは、自身のクロックの同期化の際に、すべてのローカル・サーバーの応答と 1 つのグローバル・サーバーの応答を使用します。クーリエは、以下の手順によってネットワーク全体の同期化を提供します。

1. クーリエは、リモート・エリアの少なくとも 1 つのグローバル・サーバーに対して時刻値を要求し、 **minservers** 属性で決められた数までのローカル・サーバーに対して値の差を要求します。
2. クーリエはグローバル・サーバーの時刻とローカル・サーバーの時刻を使用して、それぞれのシステムのクロックを同期化させます。
3. クーリエは、将来の同期の際に、 LAN 上の他のサーバーとクラークに新たに計算されたクロック時刻を中継します。

複数の LAN または 2 地点間リンクを含むネットワークの場合、それぞれの LAN またはセグメント上の 1 つのサーバーをクーリエとして構成する必要があります。このように構成すると、ネットワークのいろいろな部分で同期が保たれ、それらが互いに分離されることはなくなります。

管理インターフェースを使用することにより、 1 つまたは複数のサーバーをバックアップ・クーリエにするように指定することもできます。このようなローカル・サーバーは、利用可能なクーリエ・サーバーがその LAN 上にない場合に、一時的にクーリエ機能を引き受けます。このような場合、クーリエが再び有効になるまで、最低順位の汎用固有 ID (UUID) を持つバックアップ・クーリエが定期的にグローバル・サーバーと同期をとります。

グローバル・サーバーを見つけることができない場合、クーリエは、ローカル・サーバーを使用し、その **no global server detected** カウントを増やします。

第24章 DTS の実装の計画

DTS ソフトウェアと対話する人々は、システム管理者とアプリケーション・プログラマーの 2 つのカテゴリーに分けられます。プログラマーは DTS の実装の計画段階には通常関与する必要はありません。ただし、UTC 時刻のソースをサービスにインポートするプログラムを書いている場合は、プログラマーに最も近いサーバーにタイム・プロバイダーを置くことができます。タイム・プロバイダーがすぐ近くにあれば、プログラマーがソフトウェア・アプリケーションをタイム・プロバイダーのハードウェアでテストするときに便利です。

DTS の実装は、システム管理者あるいはネットワーク設計者が計画するのが普通です。どのノードをサーバーにし、どのノードをクラークにするか、また、 DTS 実装システムをネットワークとともにどのように拡張するかも、これらの人々が決定します。 DTS は大規模ネットワークに対応できるので、実装システムを拡張して新しいノードを組み込むのは比較的簡単です。

システム管理者は、ソフトウェアのインストールや DTS の保守も行います。システム管理者は、また、ネットワークの拡大に伴って、サービスが受け入れ可能な精度でもって実行されることを保証するとともに、新しいサーバー、タイム・プロバイダー、およびクラークをインストールします。

計画の全般的指針

DTS の実装を計画する際には、以下の質問について考えてください。

- セルは、単一 LAN、拡張 LAN、WAN、または LAN と WAN の組み合わせのうちのどれですか?
- 現行の、または提案するネットワーク・トポロジー (コンポーネントの配置) はどのようなものですか ?
- 必要なサーバーの数は? それらの配置場所は?
- グローバル・サーバーは必要ですか? それらの配置場所は?
- グローバル・サーバーを使用する場合、クーリエを構成する必要はありますか?
- 協定世界時 (UTC) を取得するために外部タイム・プロバイダーを使用しますか?

以下のセクションは、これらの質問に答えるために役立ちます。

DTS の計画に影響を与えるネットワーク・コンポーネントは多数ありますが、ネットワーク構成やネットワーク内のノードの数に関係なく、いくつかの一般的な規則が適用されます。これらのガイドラインを要約すると、以下のようになります。

- DTS は他の DCE コンポーネントと一緒にインストールする必要があります。
- 可能な限り、DTS サーバーを他の DCE コンポーネントのサーバーと同じノードに配置します。
- それぞれのセルには最低 3 つの DTS サーバーが必要であり、できれば、余裕をもたせるために 4 つが望ましい。
- それぞれの LAN には少なくとも 1 つのサーバーが必要です。
- サーバーは、ノードの数が最も多い場所に配置します。

ネットワークの計画の際には他の要因も考慮する必要がありますが、他の要因はネットワークのトポロジーや構成によって異なります。以下のセクションでは、ユーザー自身のネットワークに DTS を実装する際に役立つように、いくつかの代表的なセルの配置を示します。

1 つの LAN の場合の DTS の構成

ノードが 単一の LAN にある場合は、ノードの数に関係なく、 DTS の実装の計画は比較的簡単です。フォールト・タイム・サーバーを検出するために、少なくとも 3 つのシステムをサーバーとして構成します。 DTS の実装システムに余裕を持たせたい場合は、ネットワークに 4 つ以上のサーバーをインストールするように計画します。こうすれば、サーバーの 1 つで障害が起こった場合でも、 DTS から信頼性のある同期結果が得られます。

DTS 実装システムの信頼性を確実にするには、サーバー・ノード間のネットワーク接続が安定するようにします。 LAN に WAN リンクを追加する場合は、通常、WAN リンクの信頼性が LAN よりも低いため、リモート・ノードにサーバーを移動しないでください。

単一の LAN の場合は、LAN でのサーバーの位置は重要ではありません。障害追及を行いやすくするために、簡単にアクセスできるノードにサーバーの 1 つを配置すると良いでしょう。ただし、これ以外に推奨するサーバーの配置はありません。グローバル・サーバーもクーリエも必要ありません。

タイム・プロバイダーを使用する予定の場合は、起動や保守を行いやすくするために、簡単にアクセスできるシステムにタイム・プロバイダーを配置します。同期されたクロックだけがネットワークに必要で、 UTC などの時刻標準には密接に従う必要がなければ、タイム・プロバイダーは不要な場合があります。タイム・プロバイダーを使用しない場合は、 DCE 制御プログラム (**dcecp**) の **clock set** コマンドを使用して、週に 1 回程度、手作業で時刻を設定することをお勧めします。

図33 に、単純化した LAN の構成を示します。 LAN がこれよりはるかに大きい場合があっても、ネットワークの一部分はこの図に似ているはずです。

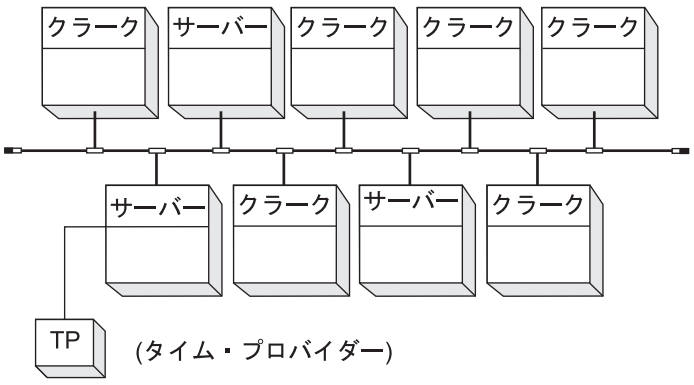


図 33. DTS 構成 - LAN

拡張 LAN の場合の DTS の構成

ネットワークがブリッジで接続された複数の LAN セグメントから構成されている場合は、ネットワークは拡張 LAN と見なされます。拡張 LAN の計画は単一の LAN の計画と似ており、拡張 LAN の各セグメントをあたかも別個の LAN のように扱います。以下のガイドラインに従うことをお勧めします。

- 各セグメントに 3 つのサーバーを作成します。
- **dts configure** コマンドを使用して、各セグメントのサーバーの 1 つをグローバル・サーバーとして構成します。
- **dts modify** コマンドを使用して、各セグメントのサーバーの 1 つをクーリエとして構成します。
- タイム・プロバイダーを使用する場合は、それらをグローバル・サーバーに接続します。

WAN および WAN リンクの場合の DTS の構成

WAN の構成にはさまざまなバリエーションがあるため、特に LAN と拡張 LAN との組み合わせの場合は、時刻を配布するために WAN リンクが使用されるケースをすべて挙げることは不可能です。このセクションでは WAN リンクが関係するすべてのケースについて推奨事項を示すことはしませんが、例としていくつかの一般的な構成を用いて、 DTS 実装システムをセットアップする方法について説明します。

WAN リンク固有の一定しない遅延のために、リンクの両端のクロック間のスキューを一定に保つことは困難です。 DTS は、WAN インターフェースを超えてクロックを同期させますが、それぞれの同期化時の伝送遅延の最悪のケースに影響されるため、クロック間に発生する誤差はより大きくなります。

WAN リンクがセルの一部である場合は、 DTS インストール・システムを常に信頼性のある堅固なものにすることが重要です。 WAN は LAN よりも信頼性が低いため、 WAN リンクを含む DTS インストール・システムは余裕を持たせるように計画します。 WAN リンクの 1 つがダウンしても、使用可能なものが常に 3 つ以上あるように、サーバーを配置してください。

以下のサブセクションでは、3 つの基本的な WAN 構成に関する推奨事項を示します。

- リモート・ノードへの WAN リンクのある、LAN または拡張 LAN
- WAN リンクによって接続されている LAN
- 中央ホスト・マシンまたはクラスターのある、すべて WAN のセル

セルがこの構成のどれとも厳密には一致しない場合でも、それぞれの例に関する推奨事項に従ってセルを計画することができます。

リモート・サイトへの WAN リンクのある LAN

282ページの図34 は、 WAN リンクを使用していくつかのリモート・ノードが組み込まれた LAN を示しています。

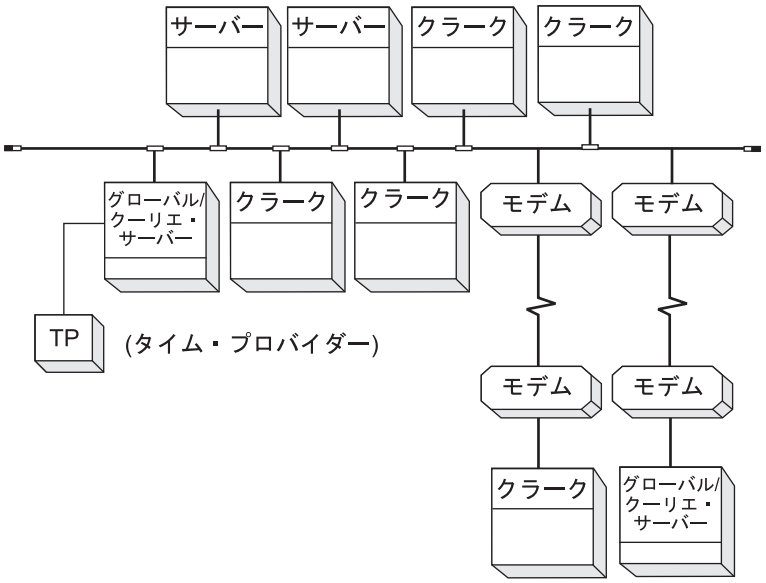


図 34. DTS 構成 - WAN リンクのある LAN

この構成では、単一の LAN に関する基本的な推奨事項に従いますが、以下の規則も守ってください。

- リモート・サイトのサーバーは、グローバル・サーバーおよびクーリエ・サーバーとして構成します。
- LAN には少なくとも 3 つのサーバーが必要です。
- 単一のタイム・プロバイダーを使用する場合は、リモート・サーバーにでなく、LAN 上のグローバル・サーバーの 1 つにそれを配置します。

前記の規則から得られるネットワーク構成では LAN 上にサーバーが集中するので、クロックのスキューは最小に保持され、サービスは、システム管理者が物理的にアクセスできない可能性のある、リモート・ノードに依存しません。各リモート・クラーク・ノードは、グローバル・サーバーと同期し、 **minservers** 属性の設定を満たします。

WAN リンクに接続されている LAN

ブリッジを使用する拡張 LAN に関して概説された規則は、 WAN リンクによって接続されている LAN にも適用されます。このようなネットワーク内の LAN は分離したエンティティーであるため、すべての LAN に対していくつかの DTS サーバーを構成する必要があります。以下のガイドラインに従って、それぞれの LAN を構成します。

- それぞれの LAN に少なくとも 3 つの DTS サーバーを構成します。
- それぞれの LAN の少なくとも 1 つのサーバーをクーリエとして構成します。
- それぞれの LAN に少なくとも 1 つのグローバル・サーバーを構成します。
- タイム・プロバイダーを使用する場合は、グローバル・サーバーにそれらをインストールします。

これらの推奨事項に従えば、WAN リンクに関連する不規則な遅延にもかかわらず、 DTS の効率と可用性は高くなります。

WAN セル

図35 は、LAN のない、地理的に分散したセルを示しています。すべて LAN の環境よりも、すべて WAN の環境の方が DTS のクロック・スキューは高くなりますが、それでも大部分の分散アプリケーションに十分な同期は提供されます。このようなネットワークでは、クロック・スキューは通常、5 秒以下ですが、衛星リンクを使用している場合は、30 秒ほどになる場合があります。

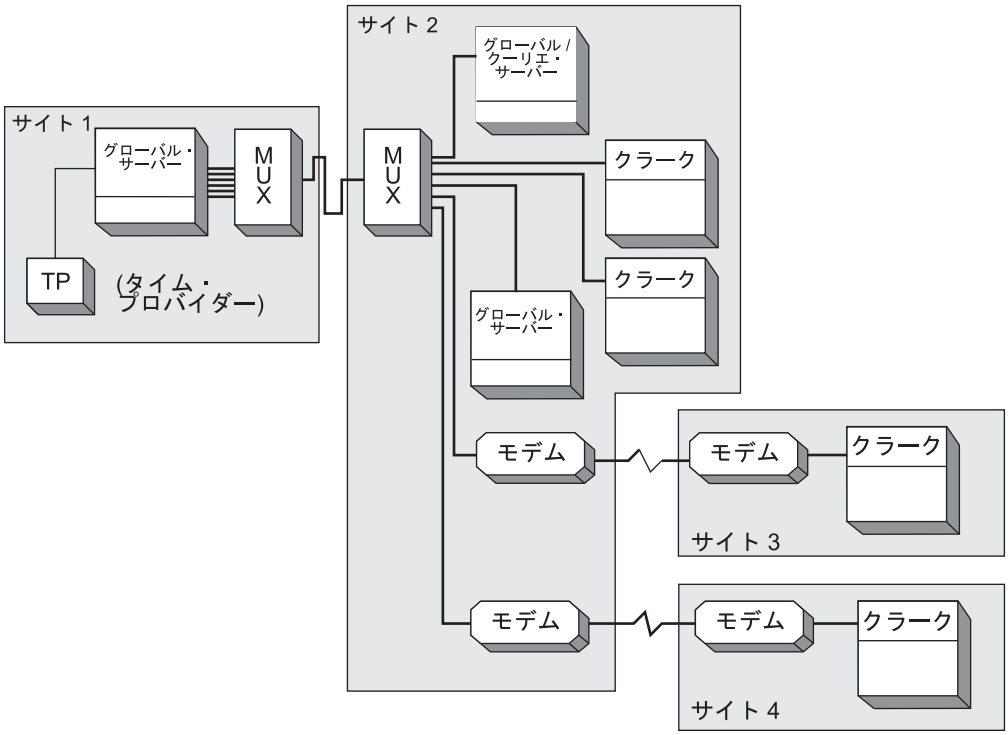


図 35. DTS 構成 - WAN ネットワーク

WAN リンクのある LAN に関する同じ推奨事項の多くが、LAN のないネットワークにも適用されます。すべて WAN のネットワークを計画する際には、以下の考慮事項に留意してください。

- ネットワークには少なくとも 3 つのサーバーが必要で、できれば 4 つ以上にします。
- すべてのサーバーをグローバル・サーバーとして構成する必要があります。
- クーリエは必須ではありませんが、サーバーの一部または全部を、クーリエとして構成することもできます。 **minservers** 属性は、少なくとも 2 つの別のグローバル・サーバーと同期をとるように、各グローバル・サーバーに強制します。
- ネットワーク内のどこでもサーバーを配置できますが、少なくとも 1 つを中央サイトに配置します。残りのサーバーに対しては、最も信頼性のあるリンクに接続された、最もアクティブなリモート・ノードを選びます。
- タイム・プロバイダーを使用する場合 (このタイプのネットワークの場合は推奨) は、 1 つを中央サイトのグローバル・サーバー・ノードに接続します。

地理的に分散した WAN ネットワークでは、システム管理者やサービス技術員がノードにアクセスしやすいことも考慮する必要があります。トラフィック・パターン

とリンクの信頼性が等しい 2 つのノードのいずれかにサーバーを配置する場合は、中央サイトや管理機能にとって便利なノードにサーバーを配置してください。

外部タイム・プロバイダーの計画

システムを UTC に緊密に同期させるために、ネットワークに 1 つまたは複数のタイム・プロバイダーを配置することができます。タイム・プロバイダーの形態は、ラジオ受信機、ソフトウェア / モデムの組み合わせ、衛星受信機など、さまざまです。 (これらのデバイスをネットワークに組み込むために使用できるタイム・プロバイダー・インターフェースの詳細については、*IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* を参照してください。 (タイム・ソースのリストについては、589ページの『付録C. タイム・プロバイダーとタイム・サービス』を参照してください。)

ネットワークでタイム・プロバイダーを使用する場合、 DTS ソフトウェアに付属している、タイム・プロバイダー・プログラムのサンプルの 1 つ (*dcelocal*/**usr/examples/dts** にある) を使用することができます。サンプル・プログラムが利用できないタイム・プロバイダーを使用する場合、あるいは特殊な要件がある場合は、タイム・プロバイダー・インターフェースに適合するタイム・プロバイダー・プログラムを書くことができます。タイム・プロバイダー・デバイスとプログラムの選択が終われば、次にネットワーク内のどこにデバイスをインストールするかを計画します。

タイム・プロバイダーを最も有利な場所に配置するのは比較的簡単です。このためには、以下のガイドラインに従ってください。

- 必ずサーバーにタイム・プロバイダーを配置します。できれば、ネットワーク内のサーバーの大多数が日常的にアクセスするサーバーにタイム・プロバイダーを配置します。
- ネットワーク構成に関係なく、可用性と有用性が最も高くなる場所にタイム・プロバイダーを配置します。
- ネットワークに複数のセグメントがある場合で、しかも、グローバル・サーバーを使用してネットワーク全体の同期を維持する場合は、グローバル・サーバー・システムにタイム・プロバイダーを配置します。

注: タイム・プロバイダーに接続したサーバーをクーリエとして構成することはできません。サーバーの処理にはタイム・プロバイダーからの時刻値だけが必要なので、タイム・プロバイダーに接続されたサーバーはクーリエの役割を引き受けません。 (クーリエ・サーバーについての詳細は、263ページの『第23章 DCE 分散タイム・サービスの紹介』を参照してください。)

第25章 DCE DTS の管理

ここでは、DCE 分散タイム・サービス (DTS) に関して行う管理作業について説明します。 DCE 制御プログラム (**dcecp**) には、これらの作業に使用できるコマンドがあります。ここには、これらのコマンドに関する簡単な説明があります。コマンドの詳細説明は、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* にあります。

dcecp が作成されるまでは、 DTS 制御プログラム (**dtscp**) が DTS の管理に使用されていました。この制御プログラムも引き続き使用できますが、その機能はすべて **dcecp** に含まれています。なお、DTS 管理のための **dtscp** コマンドの詳細説明は、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* にあります。

DCE 制御プログラムの使用法

dcecp とそのコマンド構文については、本書の 3ページの『第1章 DCE 制御プログラムの紹介』で詳しく説明しているので、ここではそれらの情報は繰り返しません。ここでは、**dcecp** が DTS 管理のために特に提供するコマンドについてだけ説明します。

DTS 用の **dcecp** コマンドは、サービスのコンポーネントを表すオブジェクトに対して各種の操作を実行します。たとえば、**dts stop** コマンドはローカル・ノード上のサーバーまたはクラークを停止します。以下のサブセクションでは、**dcecp** プログラムが操作する DTS オブジェクト、および制御プログラムがこれらのオブジェクトに対して実行できる操作のタイプについて説明します。

DTS オブジェクト

DCE 制御プログラムは、次の DTS オブジェクトを操作する機能をもっています。

- **dts**

  このオブジェクトは以下のいずれかを表します。

  – 分散コンピューティング環境内のクライアント・アプリケーションとシステムに時刻を提供する、ローカル・サーバーまたはグローバル・サーバー。

  – クライアント・システム上でクラークの役割を果たす仲介プログラム。 DTS クラークは DTS サーバーから時刻を取得してクロックを調整します。

- **clock**

  このオブジェクトは、ローカル・システムのクロックおよびそのクロックが告げる時刻を表します。

DTS のための dcecp 操作

286ページの表13 は、**dcecp** コマンドが DTS オブジェクトに行う操作の要約です。

表 13. DTS への dcecp 操作

オペレーション	説明
activate	クラークまたはサーバーのプロセスの状態を非活動状態から活動状態に変更し、オブジェクトをその時刻に同期させます。
catalog	指定されたセル内の DTS サーバーのリストを戻します。
compare	ローカル・クラークが報告した時間を、指定されたサーバーの時間と比較します。
configure	サーバーをグローバル・サーバーまたはローカル・サーバーとして構成します。
deactivate	クラークまたはサーバーのプロセスの状態を活動状態から非活動状態に変更し、オブジェクトのその時刻への同期を停止させます。
help	クラーク、サーバー、またはクロックに実行できる操作のリスト、または指定されたオブジェクトに関する長い記述を表示します。
modify	クラークやサーバーに関する属性情報を変更します。
operations	クラーク、サーバー、またはクロックに対して実行できる操作の簡単なリストを表示します。
set	クロックを徐々にまたは即座に、引き数で指定された時刻に (DTS スタイルのタイム・スタンプ・フォーマットで) セットします。
show	クラークまたはサーバーの場合、属性またはカウンターに関する情報を表示します。クロックの場合、クロックの時刻を DTS スタイルのタイム・スタンプ・フォーマットで表示します。
stop	クラークまたはサーバーのプロセスを停止します。
synchronize	**dtsd** に対して、 DTS サーバーと徐々にまたは即座に (**-abruptly** オプション) 同期をとるように命令します。

DTS オブジェクトの属性とカウンター

DTS のクラーク・オブジェクトとサーバー・オブジェクトには属性とカウンターがあります。これは一部分または一組のデータで、操作の性質を反映するかまたはこれに影響を与えます。一部の DTS クラークとサーバーの属性は、 DTS デーモンによって内部で使用され、ユーザーはその値を見ることだけができます (**dcecp dts show** コマンドにより)。他の属性には、ご使用の環境の必要性に応じてリセットできる (**dcecp dts modify** コマンドを使用して)、値が含まれています。カウンターは DTS デーモンが内部で使用します。なお、そこには見ることしかできない値が入っています。

287ページの表14 は、セットできるサーバーとクラークの属性のリストです。
287ページの表15 は、セットできないサーバーとクラークの属性のリストです。

DTS サーバーとクラークの両方の属性とカウンターについて詳しくは、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

表 14. 設定可能な DTS オブジェクト属性

サーバー	クラーク
checkinterval	--
courierrole	--
epoch	--
globaltimeout	**globaltimeout**
localtimeout	**localtimeout**
maxinaccuracy	**maxinaccuracy**
minservers	**minservers**
queryattempts	**queryattempts**
serverentry	--
servergroup	--
serverprincipal	--
syncinterval	**syncinterval**
tolerance	**tolerance**

表 15. 設定不能な DTS オブジェクト属性

サーバー	クラーク
actcourierrole	**--**
autotdfchange	**autotdfchange**
clockadjrate	**clockadjrate**
clockresolution	**clockresolution**
globalservers	**globalservers**
lastsync	**--**
localservers	**localservers**
maxdriftrate	**maxdriftrate**
nexttdfchange	**nexttdfchange**
provider	**--**
status	**--**
tdf	**tdf**
timerep	**timerep**
type	**type**
uuid	**uuid**
version	**version**

DTS タイム・スタンプ・フォーマット

dcecp コマンドに対するすべての応答には、 288ページの図36 に示す入出力フォーマットに従った、タイム・スタンプが含まれています。

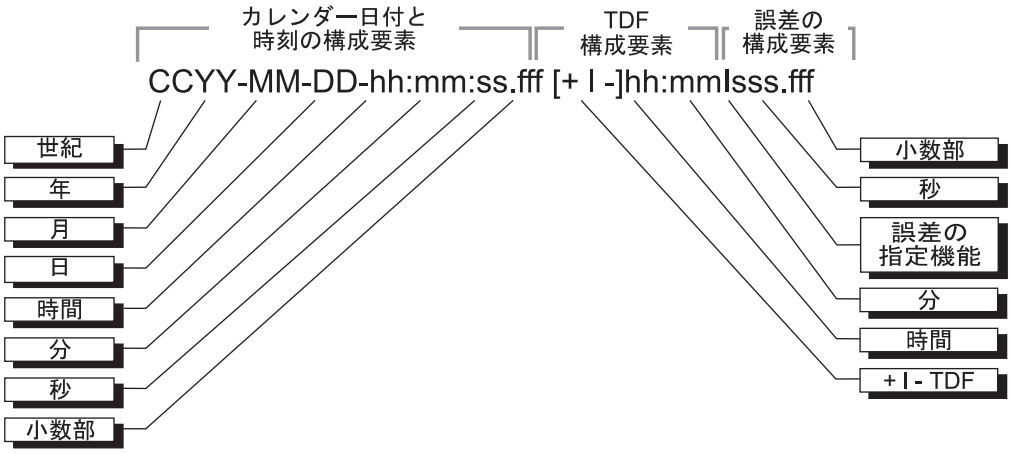


図 36. DTS タイム・スタンプのフォーマット

次の例は一般的な DTS 時刻の表示です。

```
1994-03-16-14:29:47.52000-05:00I000.003
```

タイム・スタンプは、263ページの『第23章 DCE 分散タイム・サービスの紹介』で説明されている DTS フォーマットを使用します。この例では、年号は 1994 年、日付は 3 月 16 日で、時刻は 14 時 29 分 47.52 秒です。負の時差 (TDF) の 5 時間と誤差の 3 ミリ秒がタイム・スタンプに含まれています。

ノード上の DTS の再構成

DTS は、ノードに対する DCE 全体の構成手順の中で最初は構成されます。(*IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド ― 入門* を参照)。 DCE 構成手順は、指定されたノード上に自動的に DTS サーバーと DTS クラークを作成し、活動化します。ただし、ノード上の DTS はいつでも再構成できます。再構成することを選択した場合は、以下のステップを実行します。

1. 現在そのノードで実行されているクラークかサーバーのプロセス (DTS デーモン) を停止します。
2. **dce_config** スクリプトを実行して、ノード上で DTS デーモンをクラークまたはサーバーとして再起動させます。
3. クラークまたはサーバーの属性値を必要に応じて設定します。

以下のサブセクションで、上記の再構成ステップのそれぞれについて詳しく説明します。

既存のクラークまたはサーバーの停止

ノード上の既存の DTS クラークまたは DTS サーバーを停止するには、 **dcecp dts stop** コマンドを使用します。このコマンドの実行は、まず最初にクラークまたはサーバーを非活動化し (すなわち、クラークまたはサーバーがシステム・クロックの同期をとるために使用する機能を使用不能にする)、次にプロセスを停止します。**dts stop** を次のように入力します。

```
dcecp> dts stop
dcecp>
```

dts stop コマンドは、 **dcecp dts deactivate** コマンドを呼び出して、クラークまたはサーバーのプロセスを非活動化します。これはクラークまたはサーバーのプロセスを非活動化するときに常に使用するコマンドですが、停止するコマンドではありません。 **dts deactivate** コマンドを次のように入力します。

```
dcecp> dts deactivate
dcecp>
```

新しいクラークまたはサーバーの作成

ノード上に新しいクラークまたはサーバーを作成するには、追加の DTS クラークとサーバーを構成する **dce_config** スクリプトの機能を使用します (*IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド － 入門* を参照)。追加のクラークとサーバーを構成する **dce_config** 機能は、 DCE デーモン (**dtsd**) をクラークまたはサーバーのいずれかとして再起動させます。

初期 DTS 構成時と同様、サーバーを作成する場合は、 **dce_config** スクリプトに作成するサーバーのタイプ (グローバルまたはローカル) を指定する必要があります。サーバー・タイプを選択する前に、そのサーバーがネットワーク時刻の伝搬に果たす役割を検討する必要があります。

ローカル・サーバーは、非クーリエの役割 (**courierrole** 属性の値が **noncourier** にセットされている) をもつことができます。非クーリエ・サーバーは時刻の伝搬には参加しません。ローカル・サーバーは、クーリエの役割 (**courierrole** 属性の値が **courier** にセットされている) または、バックアップ・クーリエの役割 (**courierrole** 属性の値が **backup** にセットされている) をもつこともできます。クーリエ・サーバーの第一の責任は、ネットワークのセグメント内のノード間のクロックの同期をとることです。バックアップ・クーリエは 2 次リンクであり、クーリエ・サーバーが使用できないときに時刻の伝搬を行います。ローカル・サーバーを作成すると、クーリエの役割は自動的に **backup** にセットされます。

グローバル・サーバーは **noncourier** の役割を果たさなければなりません。これをクーリエとかバックアップ・クーリエとして指定することはできません。

299ページの『グローバル・サーバーとクーリエ・サーバーの指定』 に、サーバー・クーリエの役割と、サーバーの作成後にクーリエの役割を変更する方法についての詳細が示されています。

クラークとサーバーの属性値のセット

ノード上に新しいクラークまたはサーバーを作成した後、エンティティーの属性値のうちのあるものをセットしたい場合があります。

ノードをサーバーに再構成する場合は、新しく作成されるサーバーのエポック (**epoch** 属性値) を、そのネットワーク・セグメント内に以前から存在しているサーバーが共有する、エポックと一致させる必要があります。これを行うのは、新しいサーバーが即時に既存のサーバーと同期をとれるようにするためです。サーバーのエポック番号の変更方法は 301ページの『サーバー・エポックの一致』 に示されています。

サーバーのみに適用される残りの属性をチェックし、それらが既存のサーバーの属性の設定値を補完することをチェックしたい場合があります。たとえば、サーバー

に外部のタイム・プロバイダーがある場合、 **checkinterval** 属性をチェックしたい場合があります。この属性は、 LAN 上の他のサーバーと同期をとる前にサーバーが待機する、時間の長さを指定します。

サーバーを作成した後で気が変わってサーバーのクーリエの役割を変更したい場合は、 **courierrole** 属性値を変更します。

クラークを作成した場合は、新しいクラークの属性値を、ネットワーク内の既存のクラークとサーバーの属性値と比較してチェックしたい場合があります。

DTS クラークと DTS サーバーの属性変更に関する一般的な指示は、 291ページの『クラークとサーバーの属性の変更』に記述されています。

DTS の一時的な再構成

時々、1 つまたは複数のノードの DTS の一時的な再構成を必要とする、状態あるいは問題が発生することがあります。たとえば、 LAN 内のノードに問題があり、原因となっているノードのクラークまたはサーバーの役割を別のノードに引き継ぐ必要がある場合などが考えられます。不要なサーバーやクラークをネットワークに追加するのでなく、必要な役割を果たすようにクラークまたはサーバーを変換 することができます。

クラークまたはサーバーを変換しても、その変更は一時的なものに過ぎません。ノード上で DCE が停止され、再始動されると、そのノードは初期の DTS 構成に戻ります。初期に DTS サーバーとして構成されていたノードはサーバーになり、初期に DTS クラークとして構成されていたノードはクラークになります。ノード上の DTS 構成を永久的に変更するには、 **dce_config** スクリプトを実行する必要があります。これについては 288ページの『ノード上の DTS の再構成』に説明されています。

クラークを一時的にサーバーに変換する (あるいはその逆) には、以下の手順を実行してください。

1. 現在そのノードで実行されているクラークまたはサーバーのプロセスを、**dcecp dts stop** コマンドを使用して停止します。

   ```
   dcecp> dts stop
   dcecp>
   ```

   クラークまたはサーバーを停止後、**dcecp** を終了します。
2. 該当するオプション (クラークの場合は **-c** オプション、サーバーの場合は **-s** オプション) を指定した **dtsd** コマンドを実行し、 DTS デーモンをノード上でクラークまたはサーバーとして再起動させます。たとえば、ローカル・サーバーを作成するには、次のコマンドを入力します。

   ```
   dtsd -s
   ```

   例のコマンドは、バックアップ・クーリエであるローカル・サーバーを作成します (サーバーの **courierrole** 属性値はデフォルトにより **backup** にセットされる)。必要に応じて、コマンドの **-k** オプションを使用し、 **dtsd** コマンド行でサーバーに別のクーリエの役割を指定することもできます。バックアップ・クーリエ以外に、この例で作成したローカル・サーバーは、クーリエ (**courier**) にすることも、クーリエの役割をもたせない (**noncourier**) こともできます。

次の例では、ローカル・サーバーにクーリエの役割が与えられます。

```
dtsd -s -k courier
```

グローバル・サーバーを作成するには、 **dtsd** コマンドに **-g** オプションを指定して入力します。

```
dtsd -s -g
```

注: 以前に DTS クラークを実行していたノードを DTS サーバーを実行するように再構成する場合、追加の手順を実行する必要があります。 DCE セキュリティー・サービス・レジストリー内に、新しいサーバーのためのプリンシパル・アカウントを作成し、既存の DTS サーバー・グループにサーバーの名前を追加する必要があります (**dts-entity**)。 こうしなければ、DTS クラークは新しく作成したサーバーを検出できません。プリンシパル・アカウントの作成方法については、当ガイドの 395ページの『第31章 アカウントの作成および保守』を参照してください。プリンシパル名をグループに追加する方法については、 357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。

3. DCE 制御プログラムの **dts modify** コマンドを使用して、クラークまたはサーバーの属性値を必要に応じてセットします。次のセクションでは、DTS クラークとサーバーの属性を変更する方法について説明します。

クラークとサーバーの属性の変更

多くの管理作業には DTS クラークと DTS サーバーの属性の変更が伴います。 DCE 制御プログラムには、これらのエンティティーの属性を表示および変更するためのコマンドがいくつかあります。

DTS クラークまたは DTS サーバーの属性値を表示するには、 **dts show** コマンドを使用します。 (**dts show** コマンドは、 DTS エンティティー・カウンターの値の表示にも使用できますが、カウンターの値を変更することはできません。)

たとえば、ローカル・ノード上のすべてのクライアントとサーバーの属性値を表示するには、次のコマンドを入力します。

```
dcecp> dts show
{checkinterval +0-01:30:00.000I-----}
{epoch 0}
{tolerance +0-00:10:00.000I-----}
{tdf -0-05:00:00.000I-----}
{maxinaccuracy +0-00:00:00.100I-----}
{minservers 3}
{queryattempts 3}
{localtimeout +0-00:00:05.000I-----}
{globaltimeout +0-00:00:15.000I-----}
{syncinterval +0-00:02:00.000I-----}
{type server}
{courierrole backup}
{actcourierrole courier}
{clockadjrate 10000000 nsec/sec}
{maxdriftrate 1000000 nsec/sec}
{clockresolution 10000000 nsec}
{version V1.0.1}
{timerep V1.0.0}
{provider no}
{autotdfchange no}
{nexttdfchange 1994-10-30-06:00:00.000+00:00I0.000}
```

```
{serverprincipal hosts/gumby/self}
{serverentry hosts/gumby/dts-entity}
{servergroup subsys/dce/dts-servers}
{status enabled}
{uuid 000013ed-000b-0000-b8ef-03a4fcdf00a4}
dcecp>
```

この表示例はローカル・ノード上にある 1 つのサーバーの属性値を示しています。**dts show** コマンドがクラーク用に表示する属性はこれとは異なります。また、サーバーの場合にはより多くの属性が表示されます (287ページの表14 および 287ページの表15 を参照してください)。

DTS クラークまたはサーバーの属性を変更したい場合は、 **dcecp dts modify** コマンドを使用できます。このコマンドの例をいくつか以下の項で示し、クラークとサーバーに関して設定可能な属性について説明します。それらの項では、ネットワーク構成に応じたさまざまな属性の設定値に関する提案も示します。

minservers 属性

minservers 属性は、いくつのサーバーがシステムに時刻値を提供すれば DTS がローカル・クロックの同期をとることができるかを指定します。

minservers 属性のデフォルトであり、推奨する最小値は **3** です。システムが信頼できる新しい時刻を計算するには、3 つのサーバーからの値を必要とします。サーバーかクラークかによって、システムが必要とするネットワーク内の他のシステムの数が異なります。

- クラークは 3 つのサーバーからの値を必要とします。
- サーバーは他の 2 つのサーバーからの値を必要とします。各サーバーは新しい時刻を計算する際、自分自身のクロックの値を使用します。

minservers 属性値をリセットするには、 **dts modify** コマンドに **-change** オプションを指定して、希望の値をセットします。このコマンドに指定できる値は **1** から **10** までです。たとえば、必要なサーバーの数を **4** に増やすには、次のコマンドを出します。

```
dcecp> dts modify -change {minservers 4}
dcecp>
```

localservers 属性 (LAN 内のローカル・サーバーの数を指定する) と **minservers** 属性との間に直接の関係はありませんが、 **minservers** 属性値は通常、全ローカル・サーバーのサブセットです。これらの属性の両方あるいはいずれか一方の現行値を見るには、 **dts show** コマンドを使用できます。 LAN 上の DTS ノードが最低 10 分間実行されるのを待ってから、このコマンドを出します。こうすれば、**dts show** コマンドは確実にノードの同期リストにあるすべてのローカル・サーバーを表示できます。 **dts show** コマンドは、次のように、オプションつきでも (-attributes または **-all**) 、オプションなしでも入力することができます。

```
dcecp> dts show
{checkinterval +0-01:30:00.000I-----}
{epoch 0}
{tolerance +0-00:10:00.000I-----}
{tdf -0-05:00:00.000I-----}
{maxinaccuracy +0-00:00:00.100I-----}
{minservers 4}
{queryattempts 3}
```

```
{localtimeout +0-00:00:05.000I-----}
{globaltimeout +0-00:00:15.000I-----}
{syncinterval +0-00:02:00.000I-----}
{type server}
{courierrole backup}
{actcourierrole courier}
{clockadjrate 10000000 nsec/sec}
{maxdriftrate 1000000 nsec/sec}
{clockresolution 10000000 nsec}
{version V1.0.1}
{timerep V1.0.0}
{provider no}
{autotdfchange no}
{nexttdfchange 1994-10-30-06:00:00.000+00:00I0.000}
{serverprincipal hosts/gumby/self}
{serverentry hosts/gumby/dts-entity}
{servergroup subsys/dce/dts-servers}
{status enabled}
{uuid 000013ed-000b-0000-b8ef-03a4fcdf00a4}
dcecp>
```

前の例では、**minservers** 属性値は **4** にセットされます。この設定値により、余裕が与えられます。ネットワークにグローバル・サーバーがない場合に、ローカル・サーバーが 1 つ使用不能になっても、システムは同期をとることができます。

minservers 属性の設定値で指定された数のサーバーと連絡できない場合、システムは **toofewservers** カウンターを増やし、そのイベントをログに記録し、 Too Few Servers Detected (サーバーをわずかしか検出できなかった) というイベント・メッセージを表示します。イベント・メッセージに含まれる情報に、現在有効なサーバーの数と必要なサーバーの数が示されます。このイベント・メッセージが表示された場合は、故障したサーバーがないかどうかをチェックするか、通信リンクをテストしてシステムがサーバーから分離されていないことを確認するか、あるいはネットワークにサーバーを追加します。

ネットワーク構成に応じて、 **minservers** 属性を他の方法で使用することができます。次の場合を考えてみましょう。

- ネットワークにシステムがわずかしかなく、サーバー・ドリフトに関係なく、ノードの同期をとりたい場合は、 **minservers** 属性値を **1** または **2** に下げます。この場合、結果の同期時刻は UTC の測定値より信頼性は劣りますが、システムの同期がとれる可能性は高くなります。しかし、設定が **3** 未満の場合は、システムはフォールト・サーバーを識別することができません。それ以後にサーバー・クロックのドリフトがあると UTC からの差が増大します。
- 耐障害性を高め、システムが信頼性の高い時刻を計算することを確実にするには、 **minservers** 属性値を **3** (デフォルトの設定) またはこれ以上にセットしてください。この場合、システムはフォールト・サーバーを識別し、受信した時刻値の最も狭いオーバーラップ・インターバルを計算することができます。ただし、少なくとも 3 つのサーバーが使用可能になるまで、システムは同期をとらないということを覚えておいてください。

ネットワーク内のノードの数および使用するアプリケーションのタイプによって、同期の保証または信頼できる時刻と耐障害性のどれがより重要かどうかが決まります。

グローバル・サーバーでの minservers 属性の使用

ネットワークが複数の LAN で構成されている場合、一組のグローバル・サーバーが必要です。グローバル・サーバーは、ローカル・サーバーをセル・プロファイルに公示することによって作成することができます (詳細については、300ページの『グローバル・サーバーの公示』を参照してください。)

ネットワークの中にグローバル・サーバーがある場合は、 **minservers** 属性に対して選択する値に影響することがあります。クラークまたはサーバーが利用できるローカル・サーバーの数が **minservers** 属性の設定値より小さい場合、クラークまたはサーバーはグローバル・サーバー名を求めてセル・プロファイルを自動的に検索します。その後、クラークまたはサーバーはグローバル・サーバーおよびローカル・サーバーに対して時刻値を要求します。

グローバル・サーバーが存在するかどうかは、 **dts show** コマンドを入力して **globalservers** 属性値を見ればチェックできます。 **dts show** コマンドは、次のように、オプション (**-attributes** または **-all**) を指定して、またはオプションなしで入力することができます。

```
dcecp> dts show
{checkinterval +0-01:30:00.000I-----}
{epoch 0}
{tolerance +0-00:10:00.000I-----}
{tdf -0-05:00:00.000I-----}
{maxinaccuracy +0-00:00:00.100I-----}
{minservers 3}
{queryattempts 3}
{localtimeout +0-00:00:05.000I-----}
{globaltimeout +0-00:00:15.000I-----}
{syncinterval +0-00:02:00.000I-----}
{type server}
{courierrole backup}
{actcourierrole courier}
{clockadjrate 10000000 nsec/sec}
{maxdriftrate 1000000 nsec/sec}
{clockresolution 10000000 nsec}
{version V1.0.1}
{timerep V1.0.0}
{provider no}
{autotdfchange no}
{nexttdfchange 1994-10-30-06:00:00.000+00:00I0.000}
{serverprincipal hosts/gumby/self}
{serverentry hosts/gumby/dts-entity}
{servergroup subsys/dce/dts-servers}
{status enabled}
{uuid 000013ed-000b-0000-b8ef-03a4fcdf00a4}
dcecp>
```

dts show は、ローカル・ノードが認識しているすべてのグローバル・サーバーの名前、ノード ID、およびノード名を表示します。

2 地点間回線上のシステムでの minservers 属性の使用

2 地点間 WAN リンクを介して LAN に接続するシステム上で DTS を使用している場合、単一システムでは 1 つのローカル・サーバーしか使用可能になりません。このようなシステムの **minservers** 属性に推奨する設定値は、 **3** です。クラークとして構成されたシステムの場合は、ローカル・サーバーがまったくないので、同

期のために 3 つのグローバル・サーバーに照会する必要があります。サーバーとして構成されたシステムの場合は、同期のために 2 つのグローバル・サーバーに照会しなければなりません。

maxinaccuracy 属性

maxinaccuracy 属性は、 DTS がシステムに同期を行わせるようになる、システムの誤差の最大限界を指定します。システムが **maxinaccuracy** 属性の設定値によって決められた限界を超えると、 DTS は、誤差が設定値以下のレベルに低下するまで、システムに強制的に同期化を行わせます。 **maxinaccuracy** 属性の設定値は同期化のトリガーとして使用します。システム間同期の許容度を変更するように設定値を変更することができますが、設定値を低くするとそれだけネットワークのオーバーヘッドが増えることに注意してください。デフォルトの設定値は 0.10 秒 (100 ミリ秒) です。

maxinaccuracy 属性の設定値がシステムの同期の動作に与える影響を以下に示します。

- システムのクロック値が **maxinaccuracy** 属性値より多くの誤差を累積すると、DTS が同期化を開始します。
- DTS が新しい時刻値を計算します。
- DTS がシステム・クロックを調整します。
- 新しいクロック設定値が依然として **maxinaccuracy** 属性値を超えている場合、またはその後のクロック・ドリフトにより誤差がこの値に達した場合は、このサイクルが繰り返されます。

同期を何度繰り返しても **maxinaccuracy** 属性値以下に誤差が下がらない場合、システムが連続して同期化を行う場合があります。 (『syncinterval 属性』に、**syncinterval** 属性によってこのようなループを防止する方法が示されています。)

デフォルトの **maxinaccuracy** 属性値は、ネットワーク通信またはシステム処理に介入しなくても、大部分のアプリケーションに対してシステムを十分正確に保てるように、指定されています。ネットワークに 1 つまたは複数のタイム・プロバイダーがあり、それによって非常に低い誤差に保たれている場合は、 **maxinaccuracy** 属性値を低くすることができます。次の場合にはこの値を上げてください。

- タイム・プロバイダーがネットワーク内で使用されない場合
- システムが WAN 専用ネットワーク構成の一部である場合
- DTS を呼び出すアプリケーションが、デフォルト設定値によって達成される精度のレベルを必要としない場合

次の例は、**maxinaccuracy** 属性値を 0.2 秒に変更する方法を示しています。

```
dcecp> dts modify -change {maxinaccuracy
00-00:00:00.200}
dcecp>
```

syncinterval 属性

syncinterval 属性は、システムの同期化が指定されたインターバルより頻繁に起こることがないようにします。この属性は、**maxinaccuracy** 属性が継続的に同期化を

起こすのを防止します。 295ページの『maxinaccuracy 属性』に記述されているように、 **maxinaccuracy** 属性は、システムの誤差が指定された値を超えている限り、システムの同期化を引き起こします。 **syncinterval** 属性は、指定されたインターバル値より頻繁に同期化が行われないようにします。 (**syncinterval** 属性値は、複数のシステムが同時に同期化されるのを防ぐためにランダム化されており、また、この値は正確な値ではなく、平均値です。)

maxinaccuracy 属性と **syncinterval** 属性は相互依存関係にあり、次に示す*両方*の条件が起こった場合に、自動的にシステムの同期化が起こります。

- クロックの誤差が **maxinaccuracy** 属性値以上である。
- 最後の同期化以降の時間が **syncinterval** 属性値 (多少ランダム化されている) 以上である。

システムが **syncinterval** 属性設定値に達したものの、まだ **maxinaccuracy** 属性設定値には達していない場合、システムは同期化を行わない点に注意してください。

デフォルトの **syncinterval** 属性値はサーバーの場合は 2 分、クラークの場合は 10 分です。システム間のスキューを最小化したい場合、 **syncinterval** 属性値を低くすることができます。たとえば、誤差が 100 ミリ秒に達した場合にクラークが 5 分ごとに同期をとるようにするには、次のコマンドを入力します。

```
dcecp> dts modify
-change {syncinterval 00-00:05:00.0000}
dcecp>
```

syncinterval 属性は **clock synchronize** コマンドが働くのを妨げません。このコマンドを入力すれば、いつでもシステムの同期をとることができます。
syncinterval 属性は、 **maxinaccuracy** 属性によって起動される自動同期化にのみ影響します。 (詳細については、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

tolerance 属性

tolerance 属性は、システム・クロックがフォールトになった場合の DTS の反応の仕方を決定します。フォールト・クロックは珍しい状態ですが、その原因の中には次のようなものが含まれます。

- クロック・ハードウェアの障害 (メーカーの仕様より大きいクロック・ドリフトを含む)。
- タイム・プロバイダーの誤動作。
- オペレーティング・システムによるハードウェア・クロック・チックの喪失。
- クロック値を含んだシステム・メモリーの破壊。

同期プロセス時に、クロック値とその誤差が、同期に使用しているサーバーのクロック値および誤差と交差しない場合、 DTS はシステムのクロックに欠陥があることを検出します。このプロセスを 297ページの図37 に示します。ここでは、値 **t2** がフォールトです。

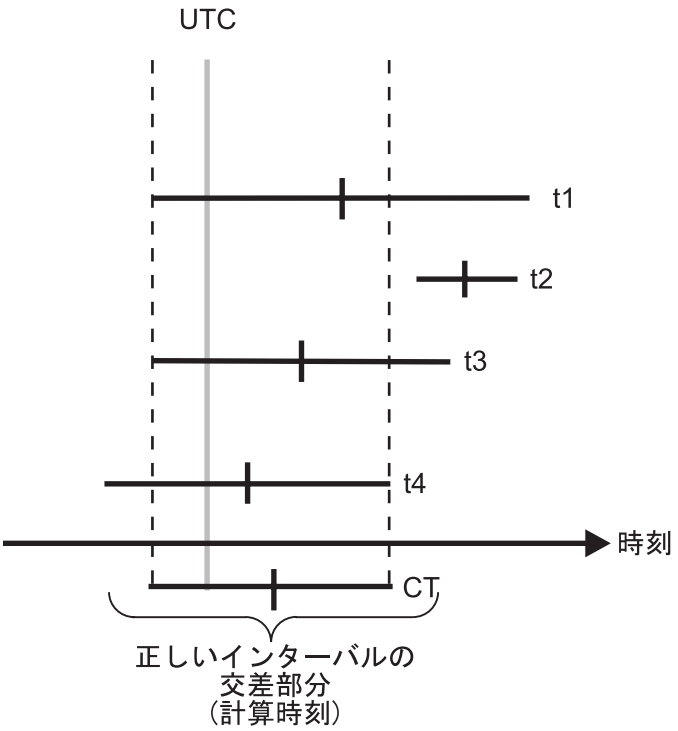


図 37. ローカル・フォールト

DTS が同期化時に欠陥のあるシステム・クロックを検出した場合、フォールトの重大度とシステムの **tolerance** 属性設定値によって DTS の反応の仕方が決まります。フォールトが検出されると、DTS は次の操作の中の 1 つを実行します。

- クロックが提供した欠陥のある時間間隔がエラー許容範囲内の場合は、 DTS はそのクロックが供給する値の誤差を増やして、クロックを徐々に調整します。
- クロックが提供する欠陥のある時間間隔がエラー許容範囲外の場合は、 DTS は直ちにそのクロックを新しい計算結果時刻に設定します。

tolerance のデフォルト設定値 (5 分) を変更する前に、システム時刻を使用するアプリケーションの要件を判別してください。 CDS サーバーのような一部の分散アプリケーションでは、システムの誤差が 5 分を超えないという条件があります。エラー許容範囲が大きいと、このようなアプリケーションが CDS ネームスペース・エントリーを正しく順序付けすることができない可能性があります。このようなアプリケーションの場合は、 **tolerance** 属性値を 5 分以下に設定する必要があります。

一部のアプリケーションでは、DTS がシステム・クロックを徐々に、かつ単調に (順方向に) 調整する必要がある場合があります。このようなアプリケーションの場合、致命的なエラーが起こった場合にのみ、クロックが即座にセットされるようにするために、 **tolerance** 属性の設定を増やすことができます。かりに **tolerance** 属性値を無限大に設定できれば、クロックが即座にセットされることが決してないことが保証されます。このような設定値は不可能ですが、10675199-00:00:00.000 (約 29,227.5 年) 未満であれば、任意の設定値を入力することができます。

次の例は、**tolerance** 属性値を 3 分にセットする方法を示しています。

```
dcecp> dts modify -change {tolerance 00-00:03:00.000}
dcecp>
```

localtimeout 属性、globaltimeout 属性、および queryattempts 属性

サーバーに照会するとき、システムは、 **localtimeout** 属性または **globaltimeout** 属性によって指定された期間だけ、応答を待ちます。 **localtimeout** 属性の設定値は、システムがローカル・サーバーと連絡しようとするときに適用され、**globaltimeout** 属性の設定値は、システムがグローバル・サーバーに連絡しようとするときに適用されます。

queryattempts 属性は、 DTS がタイムアウト・タイマーを何回リセットするとシステムが所与のサーバーとの連絡の試行をやめるかを決定します。タイムアウト設定値が **queryattempts** 属性によって決められた回数を経過した後、システムはサーバーへの照会をやめます。システムがグローバル・サーバーを照会している場合、 DTS は **Server Not Responding** イベント・レポートを生成し、そのサーバーをシステムのグローバル・サーバー・リストから削除します。 **minservers** 属性設定値を満たすためにグローバル・サーバーまたはローカル・サーバーからの応答が必要な場合、 DTS は、**Too Few Servers** イベント・レポートを生成し、システムは同期化を行いません。

queryattempts 属性のデフォルト設定値は **3** です。次の例は、**queryattempts** 属性値を **4** にセットする方法を示しています。

```
dcecp> dts modify -change {queryattempts
4}
dcecp>
```

localtimeout 属性のデフォルト設定値は 5 秒で、 **globaltimeout** 属性のデフォルト設定値は 15 秒です。グローバル設定値の方が大きいのは、グローバル・セットへのアクセスにしばしば使用される、 WAN リンク上での通信遅延を考慮しているためです。 **localtimeout** 属性の設定値を変更する必要はまずありません。しかし **globaltimeout** 属性の設定値は、 WAN のトポロジーと伝送品質が多様なので、変更が必要になる可能性があります。次の例では、 **globaltimeout** の設定値は 20 秒に変更されます。

```
dcecp> dts modify -change {globaltimeout 00-00:00:20.000}
dcecp>
```

グローバル・サーバーについての **Server Not Responding** イベント・レポートを連続して受信した場合は、**globaltimeout** の設定値を大きくしてください。設定値を大きくしてもイベント・レポートが継続する場合は、サーバーとの通信リンクに問題がある可能性があります。

serverentry 属性と serverprincipal 属性

DCE および DTS の初期構成時に、 CDS で使用するために 1 つの DTS エントリー名が作成され、またレジストリー・サービスで使用するために 1 つの DTS 名が作成されます。その後でサーバーの名前を変更したくなった場合には、サーバーの 2 つの属性、つまり **serverentry** 属性と **serverprincipal** 属性を変更すればそれができます。これらの **dcecp** プログラムの属性のデフォルト設定値は、初期 DCE 構成時に作成された名前のデフォルト設定値と同じです。*これらは推奨設定値*です。この節では、このような属性の設定値に関してさらに考察します。

serverentry 属性と **serverprincipal** 属性の設定値を変更することに決めた場合は、新しい値が適切であることを確認してください。適切でない場合は、DTS に問題が生じます。

serverentry 属性は、サーバーのバインディングがエクスポートされる、CDS エントリー名を指定します。この属性の設定値を変更すると、ネームスペースのエントリーも変更されます。次の例は、**serverentry** 属性値をセットするコマンドの例です。

```
dcecp> dts modify  -change {serverentry /.:/hosts/cyclops/dts_ref_node}
dcecp>
```

serverprincipal 属性は、認証に使用されるサーバーのプリンシパル名を指定します。 **dcecp** を使用して名前を変更する場合は、合致するプリンシパル名とアカウントをセキュリティー・サービス・レジストリーに作成する必要があります。これを行うとき、新しいプリンシパル名を既存の DTS サーバー・グループ (**dts-servers**) に加えなければなりません。マシン・プリンシパルはこの許可グループのメンバーでなければなりません。新しいプリンシパル・アカウントの作成に関する詳細については、当ガイドの 395ページの『第31章 アカウントの作成および保守』を、既存のサーバー・グループにプリンシパル名を追加する方法については、357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。

次のコマンドの例は、**serverprincipal** 属性をセットします。

```
dcecp> dts modify -change {serverprincipal /.:/hosts/ajax/dts_machine}
dcecp>
```

サーバー特有の管理作業

DTS サーバーの管理にはいくつかの特殊な作業が含まれます。これらの作業には以下のものが含まれます。

- サーバーのエポックの設定
- サーバーへのクーリエの役割の割り当て
- サーバーのグローバル・サーバーとしての指定
- タイム・プロバイダーへの接続のための属性の設定

次のサブセクションは、これらのサーバー特有の作業について説明します。

グローバル・サーバーとクーリエ・サーバーの指定

ネットワークが WAN リンクを持っているか、それが拡張 LAN である場合、別のネットワーク・セグメント内のノードの同期をとるために、グローバル・サーバーとクーリエ・サーバーを使用する必要がある可能性があります。ネットワーク全体のノードの同期をとるには、いくつかのサーバーにグローバルの役割を割り当て、選択したローカル・サーバーにクーリエの役割を割り当てます。 (グローバルおよびローカル・クーリエ・サーバーの位置の計画についての助言は、 263ページの『第23章 DCE 分散タイム・サービスの紹介』を参照してください。) サーバーの役割を割り当てるには、以下の項の指示に従ってください。

グローバル・サーバーの公示

サーバーのグローバル・セットにサーバーを割り当てるには、 **dcecp dts configure** コマンドでそのサーバーを公示する必要があります。サーバーを公示すると、同時にサーバーの CDS 名にバインディング情報が追加され、またセル・プロファイルにサーバーのエントリーが追加されます。 CDS とセル・プロファイルはネットワーク内のすべてのノードで利用可能であるため、 DTS はセル・プロファイル内を検索して、 LAN 上で到達できないノードの位置を取得することができます。

次のコマンド例は、サーバーをグローバル・サーバーとして通知し、それによってそのサーバーを CDS に登録し、それをセル・プロファイルの中に入れる方法を示しています。

```
dcecp> dts configure -global
dcecp>
```

-global オプションは、サーバーをローカル・サーバーでなく、グローバル・サーバーとして構成することを指定します。

サーバーをグローバル・サーバーとする指定を削除するには、次のように **dts configure** コマンドを使用します。

```
dcecp> dts configure -notglobal
dcecp>
```

このコマンドは、グローバル・サーバーを公示解除し、そのエントリーをセル・プロファイルから、またバインディング情報をその CDS 名から、それぞれ削除します。

サーバーへのクーリエの役割の割り当て

クーリエ・サーバーは、ネットワーク内の分離した部分にあるシステム間の同期を維持するのに重要な役割を果たします。クーリエ・サーバーは同期のたびごとに、少なくとも 1 つのグローバル・サーバーに対して時刻値を要求します。この手順によって、クーリエ・サーバーはリモート・システムの時刻を LAN またはローカル・エリアに伝搬し、それによってネットワークの他のすべての部分と LAN の同期を維持することができます。

サーバーに割り当てることができるクーリエの役割 (**courierrole** 属性) には、次の 3 つがあります。

- **バックアップ**
- **クーリエ**
- **非クーリエ**

グローバル・サーバーまたはローカル・サーバーの作成時におけるデフォルトのクーリエの役割は**バックアップ**です。

サーバーをネットワークの他の部分への 1 次リンクとして指定するには、 **courierrole** 属性の **courier** 設定値を使用します。サーバーをネットワークの他の部分への 2 次リンクとして指定するには、 **backup** 設定値を使用します。バックアップ・クーリエは LAN 上に他のクーリエがない場合にのみ有効です。

バックアップ・クーリエの役割には処理やオーバーヘッドにほとんど負担がかからないことに注意してください。 LAN のサーバーの 1 つをクーリエとして指定し、LAN のその他のサーバーをすべてバックアップ・クーリエとして指定することができます。複数のサーバーをバックアップ・クーリエとして構成した場合、クーリエが使用不能になると、最低順位の UUID を持つバックアップ・クーリエが有効なクーリエになります。

サーバーにクーリエの役割を割り当てるには、次の **dcecp** コマンドを入力します。

```
dcecp> dts modify -change {courierrole courier}
dcecp>
```

サーバーにバックアップ・クーリエの役割を割り当てるには、次のコマンドを入力します。

```
dcecp> dts modify -change {courierrole backup}
dcecp>
```

サーバー・エポックの一致

起動時には、サーバーのエポック番号は、互いに同期している他のサーバーのエポック番号と一致していなければなりません。同期をとるにあたって、サーバーはエポック番号が自分のと一致しないサーバーからのクロック値は無視します。

DTS サーバーが最初に使用可能になったとき、それぞれのサーバーのエポック番号は 0 なので、初期インストール時にはエポック番号を変更する必要はありません。その後、既存のネットワークにサーバーを追加するか、あるいはクラークをサーバーに変更する場合は、新しいサーバーと既存のサーバーのエポック番号が一致していることを確認してください。サーバーのエポック番号を検索するには、 DCE 制御プログラムの **dts show** コマンドを入力します。たとえば、以下のとおりです。

```
dts show /.:/hosts/orion/dts-server
```

コマンドが戻した属性リストを見て、そのサーバーの **epoch** 属性値を調べてください。作成したサーバーのエポックが他のサーバーのものと一致した場合、新しいサーバーは即時に同期をとることができます。しかし、エポックが一致しない場合は、新しいサーバーのエポックを変更しない限り、新しいサーバーは既存のサーバーを無視します。次の例は、サーバーを使用可能にした後でそのサーバーのエポック番号を変更する方法を示しています。

```
dcecp> clock set -abruptly -epoch 0
dcecp>
```

サーバーが適切なエポック番号で起動されたことが分かったら、重大なシステム上あるいはネットワーク上の問題によってすべてのサーバー・クロック値が壊されない限り、エポックを変更しないでください。大多数のサーバー・クロックが故障するという事態になった場合は、 **dts show** コマンドと **clock set** コマンドを使用して問題のサーバーを分離し、他の DTS アプリケーションに影響を与えずに、問題のトラブルシューティングや保守を行えるようにします。

タイム・プロバイダーに接続するための checkinterval 属性の設定

サーバーがタイム・プロバイダーと接続される場合は、その **checkinterval** 属性をセットします。 DTS は **checkinterval** 属性を使用して LAN 上のすべてのサーバ

ーを定期的に検査し、タイム・プロバイダーとの同期が保たれていることを確認します。 **checkinterval** 属性の設定値によって指定された時間量を超えた場合、タイム・プロバイダーを持つサーバー (TP サーバー) は、次に示す手順を実行します。

1. TP サーバーは LAN 上の他のすべてのサーバーに時刻値を要求します。
2. TP サーバーは同期プロセスを開始します。
3. TP サーバーは、自身の時間間隔と交差しない時間間隔を持つサーバーを識別します。
4. TP サーバーは、検出したフォールト・サーバーそれぞれについて、イベント・メッセージを出します。

前述の順序列では、 TP サーバーが同期プロセスを開始後に実際にはシステム・クロックを設定しないことに注意してください。 TP サーバーは、単にフォールト・サーバーを検出するためにプロセスを実行するだけです。 DTS ソフトウェアは、TP サーバーの時刻値が利用できる時刻の中で最も正確だという前提に立っているので、 TP サーバーは他のサーバーから集めた値を使用して自身のクロックを変更することはありません。その代わり、 TP サーバーは他のサーバーのための基準計時機能として機能します。

検査インターバルの値を低く設定して、フォールト・サーバーの通知がもっと早く行われるようにすることができますが、設定値を低くすると、ネットワーク・リソースにかかる負荷が増すことに注意してください。次の例は、**checkinterval** 属性値をセットする方法を示しています。

```
dcecp> dts modify /.:/hosts -change {checkinterval
00-00:00:30.0000]
dcecp>
```

システム時刻の変更

dcecp コマンドを使用してシステムの時刻を変更する方法は 3 つあります。以下のサブセクションでは、システム時刻の変更の理由について説明し、そして、時刻の変更とシステム・クロックの変更に使用できるコマンドの例を示します。

時刻の単調な更新

タイム・プロバイダーを使用しないネットワークの場合で、しかもネットワーク・システムがある期間すでに実行されている場合には、複数のシステム上の時刻を更新して UTC または他の外部参照時刻に合わせることができます。ネットワークにタイム・プロバイダーがなければ、システムが緊密に同期がとれていても、そのクロックは、受け入れた時刻標準 (UTC など) からずれている可能性があります。

サーバー・システム上の時刻をより正確なものにするために変更したい場合は、**dcecp clock set** コマンドを使用します。 DTS 同期プロセスによって、このコマンドにより提供された新しい時刻は確実に他のネットワーク・システムに伝搬されます。システム・クロックを新しい時刻に更新するには、システムに指定した新しい時刻と誤差から得られるインターバルが、現行のシステム・インターバルより小さくなければなりません。

clock set コマンドを効果的に使用するには、正確な参照時刻に一時的にアクセスできなければなりません。このような参照時刻には、多くの標準組織が無線や電話

によって伝搬している、時刻シグナルを含めることができます。最近正確であると確認できたクロックを使用することもできます。 (UTC 時刻のプロバイダーについては、589ページの『付録C. タイム・プロバイダーとタイム・サービス』 を参照してください。)

clock set コマンドは絶対時刻を変更するための、手動で入力するコマンドであるため、小さな誤差の設定には向いていません。このコマンドで達成できる、信頼できる最小の誤差は約 1 秒です。人的エラーと処理遅延とを組み合わせると、これより低い設定値は信頼できなくなります。たとえば、コマンドと新しい時刻を入力してから、参照のモニターを始めるとします。参照が希望の時刻に達したと認めた時点で、 **<Return>** を押すとコマンドが開始します。参照マークを読み取った時点と **<Return>** が押される時点とは完全には一致しません。さらに、コマンドが開始されてから、DTS がコマンドを解釈して実行するための時間がかかります。

次の例はサーバー・システム上で時刻を単調に更新する方法、すなわちクロックをリセットし、最終的にネットワーク全体に調整を伝搬する方法の例を示しています。

```
dcecp> clock set  1994-10-07-09:30:15.00I01.00
dcecp>
```

システムを基準 (UTC など) に対して 1 秒より厳密に同期させる必要がある場合は、 589ページの『付録C. タイム・プロバイダーとタイム・サービス』にリストされている、タイム・プロバイダーの 1 つを購入することを検討してください。 リストされているタイム・プロバイダーはすべて、伝送や処理の遅延を補正し、ミリ秒レベルの正確さで時刻参照を提供できます。

時刻の非単調な更新

サーバー・システム用の時刻を即時に設定したい場合は、 **-abruptly** オプションを指定した **clock set** コマンドを使用します。 **-abruptly** オプションを指定した **clock set** コマンドは、徐々に (単調に) 時刻を調整するのではなく、即時に (単調でなく) システム・クロック設定値を指定された時刻に変更します。

注: システム時刻を即時に変更する場合は用心してください。時刻の突然の調整は、システムを起動する時、またはシステム・クロックが故障したためにその問題を識別して訂正する場合には適しています。システム時刻の変更の結果、設定値がそのシステムの認識しているサーバーの時間間隔から外れる場合は、DTS は次回の同期の時にシステムの故障を宣言します。

clock set コマンドは、通常、著しいクロック・エラーの訂正に使用されるため、システムとサーバーのエポック番号が同じ場合、あるシステムに指定した時刻が、システムの認識しているサーバーからはフォールトであるように見える可能性があります。時刻を変更するシステムがフォールトと宣言されるのを防止することができます。新しい時刻を設定するときに **clock set** コマンドの **-abruptly** オプションといっしょに **-epoch** オプションを使用して、そのシステムを他のシステムから分離します。その上で、すべてのシステムが再び同じエポックを共用するようになるまで、他のシステムの時刻とエポックを変更しておくことができます。このプロセスは、ネットワーク内の大多数のサーバーがフォールトになるようなまれなケースの場合に便利です。

clock set コマンドを効果的に使用するには、正確な時刻参照に一時的にアクセスできなければなりません。このような参照時刻には、多くの標準組織が無線や電話によって伝搬している、時刻シグナルを含めることができます。最近正確であると確認できたクロックを使用することもできます。 (時刻参照ソースのリストについては、589ページの『付録C. タイム・プロバイダーとタイム・サービス』 を参照してください。)

clock set コマンドは絶対時刻を変更するための、手動で入力するコマンドであるため、小さな誤差の設定には向いていません。このコマンドで達成できる、信頼できる最小の誤差は約 1 秒です。人的エラーと処理遅延とを組み合わせると、これより低い設定値は信頼できなくなります。たとえば、コマンドと新しい時刻を入力してから、参照のモニターを始めるとします。参照が希望の時刻に達したと認めた時点で、 **<Return>** を押すとコマンドが開始します。参照マークを読み取った時点と **<Return>** が押される時点とは完全には一致しません。さらに、コマンドが開始されてから、DTS がコマンドを解釈して実行するための時間がかかります。

次の例は、システムの時刻とエポックを両方を変更する方法を示しています。

```
dcecp> clock set 1993-10-07-09:30:15.0000I01.0000 -abruptly
-epoch 1
dcecp>
```

システム同期化の強制

ネットワーク内にあるすべてのシステム上の DTS を作成して使用可能にすると、それ以上介入しなくてもシステムは同期をとります。ただし、 **syncinterval** 属性と **maxinaccuracy** 属性で指定された時間だけ待つのでなく、システムが直ちに同期をとるように強制したい場合もあります。一例として、ネットワークに追加したばかりの TP サーバーとシステムを同期させたい場合があります。

システムのクロックを強制的に同期させるには、**dts synchronize** コマンドを使用します。 **-abruptly** オプションを指定しない **dts synchronize** コマンドを入力すると、時刻は徐々に調整されます。 **-abruptly** オプションを指定した **dts synchronize** コマンドを入力すると、時刻は直ちに調整されます。ここで示す例で提示された状況では、 **-abruptly** オプションを使用して、タイム・プロバイダーにより提供された狭い時間間隔をネットワーク全体に早く伝搬させることができます。

```
dcecp> dts synchronize -abruptly
dcecp>
```

DTS へのアクセスの制御

DCE 許可サービスのアクセス制御リスト (ACL) を使用して、 DTS オブジェクトに対するアクセスを制御する特権を割り当てることができます。

与えられたシステム上のサーバーを表す DTS プリンシパルは、 DTS 用の 1 次アクセス制御オブジェクトです。このプリンシパルは、人間のユーザーおよび、クラーク・プロセスまたはサーバー・プロセスによるアクセスを制御してきました。すべての **dcecp** コマンドの DTS オブジェクトに使用できるデフォルト名は、**/.:/hosts/***hostname***/dts-entity** です。

DTS サーバー用の ACL には、プリンシパル (人またはプロセス) またはこのプリンシパルが属する許可グループに有効な、 ACL エントリーの任意のタイプを含めることができます。 DCE ACL 機能の説明および、 ACL タイプとそのエントリーについては、当ガイドの 329ページの『第28章 アクセス制御リストの使用』を参照してください。

DTS サーバー・プリンシパルの ACL 内の ACL エントリーは、 **dcecp acl show** コマンドを使用して表示することができます。たとえば、以下のとおりです。

```
dcecp> acl show /.:/hosts/Detroit2/dts-entity
{unauthenticated r--}
{user hosts/Detroit2/self rwc}
{group subsys/dce/dts-admin rwc}
{any_other r--}
dcecp>
```

DTS サーバー・プリンシパルの ACL 内のエントリーは、 **acl modify** コマンドを使用して変更することができます。このコマンドの使用法は、 329ページの『第28章 アクセス制御リストの使用』にあります。

第26章 ネットワーク・タイム・プロトコルとの相互運用

ネットワーク・タイム・プロトコル (NTP) は、インターネット推奨の標準です。NTP 同期サブネットワークは、時刻サーバーを表すノードと、ノード間の伝送パスを表すエッジからなる、ツリー構造のグラフによって表現されます。ツリーのルート・ノードは 1 次サーバーと呼ばれ、ラジオ放送や調整済み原子時計と同期します。残りのノードは 2 次サーバーと呼ばれ、他のサーバー (1 次および 2 次) と同期をとります。

特定のサーバーと 1 次サーバーとの間のサブネットワーク・ホップの数により、サーバーの層が決まります。つまり、ホップ数が少ないほど、層が低くなります。低い層のサーバーは、高い層のサーバーより常に高い正確度を持っています。すべてのサーバーは同じ機能を持っており、次に低い層のクライアントおよび次に高い層のサーバーとして同時に動作できます。

サーバー (1 次と 2 次の両方) は、通常、同じ層または低い層にある他のいくつかのサーバーとともに NTP を実行します。選択アルゴリズムは、ローカル・クロックを実際に同期させる元として、最も正確で信頼性のある 1 つまたは複数のサーバーを選択します。

NTP と DTS は両方とも、ローカル・ネットを組み込んだ大規模コンピューター・ネットワーク (つまり、ルーター、ルーター・ゲートウェイ、およびブリッジによって接続されたもの) で使用でき、ブロードキャスト (ブロードキャスト) と 2 地点間伝送の両メディアを使用します。 DTS と NTP は、同じ LAN で同時に実行できます。

以下のセクションでは、ローカルおよびリモートの NTP 時刻ソースとの間で時刻の受け渡しをする方法、およびループを防止する方法について説明します。

NTP 時刻ソースからの時刻の取得

DTS は、以下の 2 つのタイム・プロバイダー・プログラムを提供しています。

- **dts_ntp_provider.c** -- これは、ラジオ受信機からの場合と同様に、NTP サーバーから時刻を取得します。 ユーザーは、NTP サーバーの名前と誤差を指定します。
- **dts_null_provider.c** -- これは、NTP などの外部エージェントによってすでにそのクロックが同期化されている、 DTS サーバーで使用されます。これは誤差をセットしますが、DTS に時刻をセットさせないようにします。ユーザーは、NTP に関するその場所での経験に基づいて、誤差をセットします。ヌル・タイム・プロバイダーは、NTP によって管理されているラジオ・クロックがすでに存在するサイトでは便利な場合があります。この場合は、ラジオ・クロックのあるノードを DTS サーバーにして、ヌル・タイム・プロバイダーを使用します。

ローカル NTP 時刻ソースからの時刻の取得

NTP クロック・ドライバーを実行しているノードで、クロックとヌル・タイム・プロバイダーを使用して DTS サーバーを実行します。時刻ソース (例: ラジオ・クロ

ック) と一致する方法で誤差を指定します。他の DTS サーバーは、このソースから時刻を取得します。この場合、システムは時刻ソースと接続されているため、これは NTP 層 1 サーバーです。

以下に示す規則と注意事項を守ってください。

- 規則 - それがサブネットワーク内の唯一のローカル時刻ソース (ラジオ・クロック) の場合、それ以外の DTS ノードが NTP に時刻を与えないことを確認します。ただし、他のローカル時刻ソースがある場合は、この制限は適用されません。
- 規則 - ローカル時刻ソースがない場合、ヌル・タイム・プロバイダーを実行させないでください。
- 注意事項 - 非常に小さいポーリング率 (約 1 秒) を使用します。
- 注意事項 - NTP は **adjtime( )** システム・コールを行うため、ローカル・ノードに指定されない誤差を生じることがあることに注意してください。

図38 は、ローカル時刻ソースをもつ DTS サーバー / クライアントが、 NTP 層 1 サーバーから時刻を取得する方法を示しています。

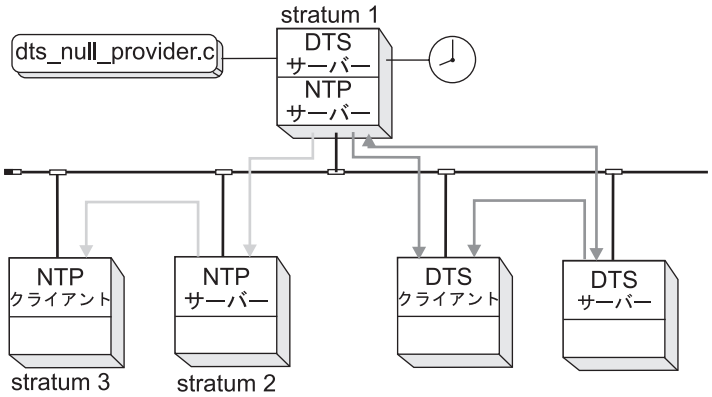


図 38. ローカル時刻ソース

リモート NTP 時刻ソースからの時刻の取得

NTP サーバーにアクセス可能なノード上で、 NTP タイム・プロバイダー (**dts_ntp_provider.c**) を使用して、DTS サーバーを実行します。 NTP に関するその場所での経験と一致する方法で誤差を指定します。

以下の注意事項を守ってください。

- 注意事項 - リモート・ソースへのリンクが遠い場合は、サブネットワーク・ノードの 1 つで NTP をローカルに実行することを考慮してください。
- 注意事項 - NTP タイム・プロバイダーは、層 8 以上の NTP ノードからの時刻は受け入れないことに注意してください。
- 注意事項 - NTP ノードは、できるだけ層 1 の近くにある必要があります。

309ページの図39 と 309ページの図40 はいずれも、リモート NTP 時刻ソース (層 3 サーバー) から時刻を取得する DTS サーバーを示しています。

ただし、図39 (シナリオ 1) では、このセクションに示した注意事項がすべて守られており、 図40 (シナリオ 2) では、最初の注意事項 (リモート・ソースへのリンクが遠い場合にサブネットワーク・ノードの 1 つで NTP をローカルに実行する) が守られていません。

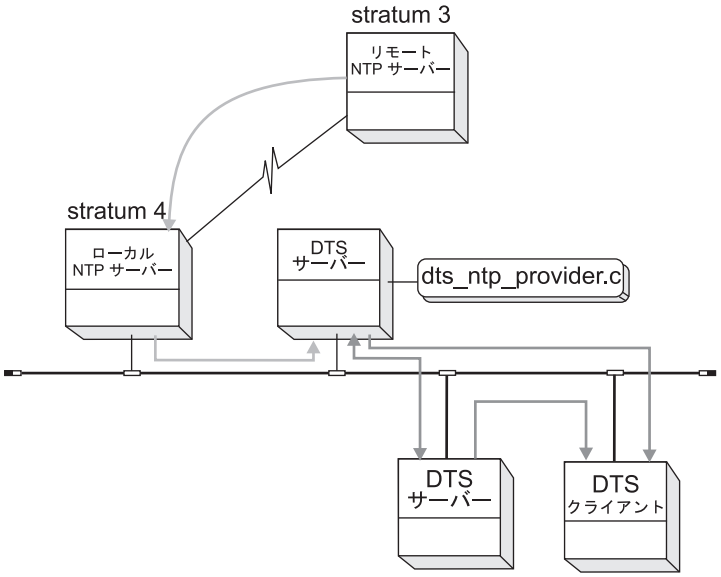


図 39. リモート NTP 時刻ソースからの時刻の取得 (シナリオ 1)

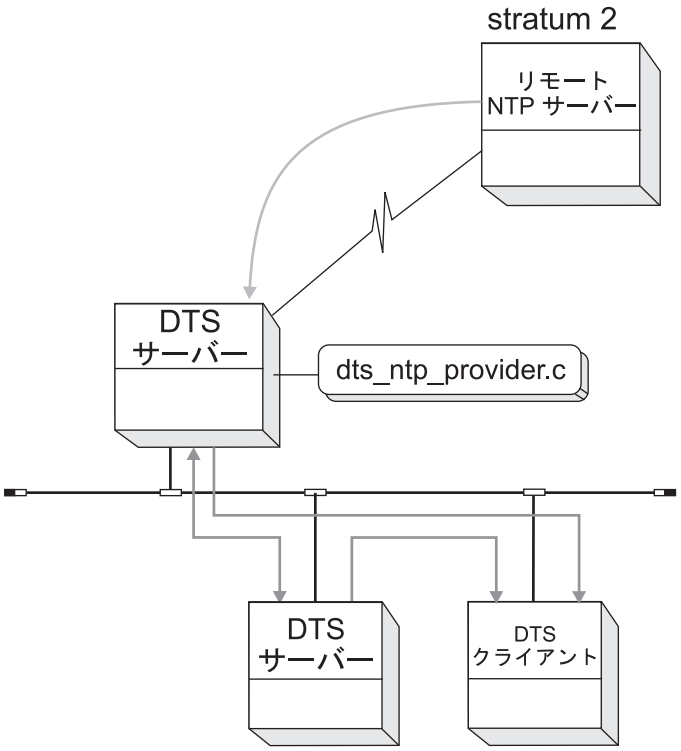


図 40. リモート NTP 時刻ソースからの時刻の取得 (シナリオ 2)

NTP ノードへの時刻の提供

-s オプションを指定し、特別な構成ファイル (**ntp.conf**) を使用して、 **ntpd** デーモンまたは **xntpd** デーモンを実行する、 DTS サーバーまたは DTS クラークは、NTP サーバーとして構成することができます。

ntpd デーモンを実行するシステムの場合、 **ntp.conf** 構成ファイルには、次に示す行が含まれていなければなりません。

```
peer /dev/null DTSS 8 -5 local
```

さらに、ファイル **/sbin/init.d/ntpd** の **ntpd** エントリーに **-s** を追加するか、あるいは、**rc.local** のあるシステムについては、**ntpd** を開始する行を変更します。

xntpd デーモンを実行するシステムの場合、 **ntp.conf** 構成ファイルに次の行が含まれている必要があります。

```
peer 127.127.1.8
```

さらに、ファイル **/sbin/init.d/xntpd** の **xntpd** エントリーに **-s** を追加するか、あるいは **rc.local** を使用しているシステムについては、 **xntpd** を開始する行を変更します。

この構成では、NTP はクロックを設定しません。ただし、NTP は他の NTP クライアントに時刻を提供することができます。 DTS と NTP の間にループを起こさないようにしてください。 NTP が DTS に時刻を渡し、その後、 DTS がその時刻を *同じ* NTP サーバーのセットに戻すと、予期しない結果が起こる可能性があります。

NTP 構成ファイルは、 DTS から時刻を取得する NTP サーバーが Stratum (層) 8 ノードになるように設定されます。さらに、**dts_ntp_provider** は、 Stratum (層) 8 以上のノードから時刻を受け入れることを禁止されています。

DTS (サーバー) ノードは、以下の規則と注意事項が守られていれば、 NTP ノードに時刻を提供することができます。

- 規則 - **ntp.conf** ファイルは、このノードを Stratum (層) 8 で宣言する必要があります。
- 注意事項 - このセット内の複数のノードが **ntpd -s** または **xntpd -s** を実行できます。
- 注意事項 - DTS 管理のシステムにローカル時刻ソースがある場合、そのシステムは NTP **-s** サーバーとして使用する必要があります。
- 注意事項 - この操作は DTS サーバー、DTS クライアントのどちらのノードでも実行できますが、 DTS サーバーの方を推奨します。

注: ヌル・タイム・プロバイダーを使用する場合、 307ページの『ローカル NTP 時刻ソースからの時刻の取得』の規則にも従う必要があります。これは、NTP ノードで実行されるヌル・タイム・プロバイダーは、層の検査をバイパスする可能性があるためです。

図41 は、**ntpd -s** を実行し、 NTP サブネットワークに時刻を提供する、2 つの DTS サーバー・ノードを示しています。 **ntp.conf** ファイルは、これらのサーバーを Stratum (層) 8 に定義します。

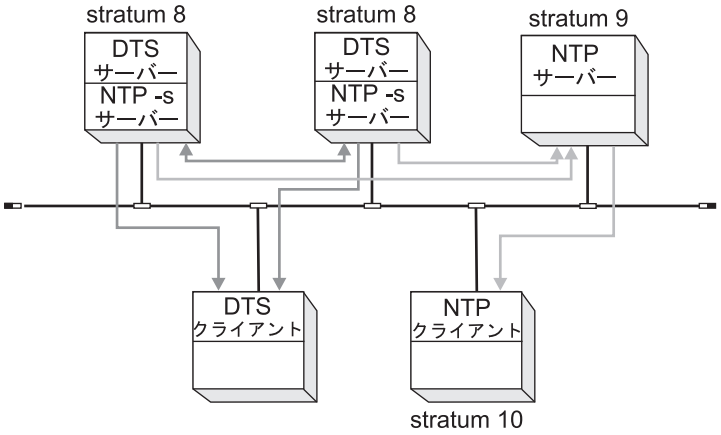


図 41. NTP への時刻の提供

ループの防止

NTP → DTS → NTP のようなループが形成されないように してください。

ヌル・タイム・プロバイダー (**dts_null_provider.c**) は、ローカル時刻ソースがある場合にだけ実行してください。ローカル時刻ソースがない場合でも、ヌル・タイム・プロバイダーを実行することができますが、ローカル・セット内のどこにも NTP の時刻を配布しないでください。

312ページの図42 は、お勧めできない 構成を示しています。この構成は、リモート NTP Stratum (層) 2 ノードに障害が起こらない場合にのみ機能します。

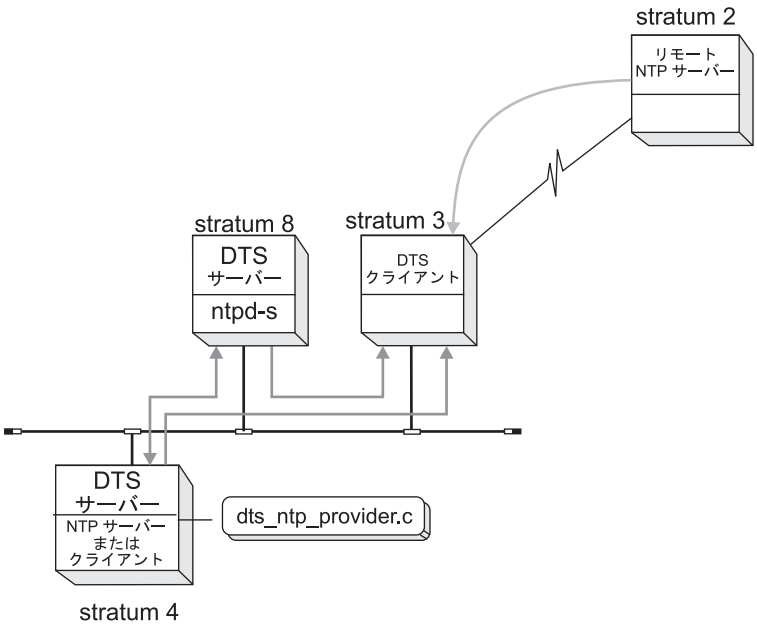


図 42. Stratum (層) 2 ノードに障害が起こる前の構成

リモート NTP Stratum (層) 2 ノードに障害が起こると、 Stratum (層) 3 ノードが Stratum (層) 8 ノードから時刻を受け入れ始めます。これが起こると、 図43 に示すように、Stratum (層) 3 ノードは Stratum (層) 9 にドロップし、 Stratum (層) 4 ノードは Stratum (層) 10 にドロップします。

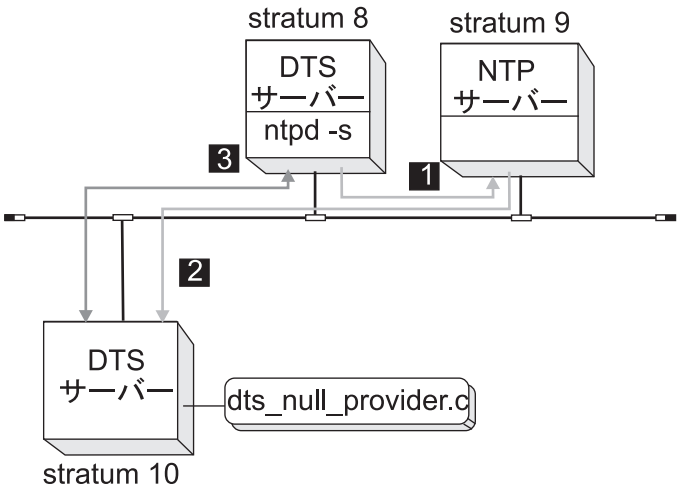


図 43. Stratum (層) 2 ノードに障害が起こった後の構成

図43 のシナリオは、以下のようなループが形成されたことを示しています。

1. Stratum (層) 8 と表示されたノードから、Stratum (層) 9 と表示された NTP ノードに進む。
2. Stratum (層) 9 と表示された NTP ノードから、Stratum (層) 10 と表示されたノードに続く。
3. それから、DTS は Stratum (層) 8 と表示されたノードに時刻をフィード・バックし、ループが形成される。

これが発生すると、NTP と DTS サブネットワークの時刻は、UTC からドリフトする可能性があります。

第6部 DCE セキュリティー・サービス

第27章 DCE セキュリティーの概要

この章の説明は、セキュリティー・レジストリーの非 LDAP 実装、つまり既存の DCE に関するものです。 DCE セキュリティー・レジストリーの LDAP 実装の概要については、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

ここでは、DCE セキュリティー・サービスについて簡単に紹介します。 DCE セキュリティー・サービスは、以下のサービスで構成されています。

- レジストリー・サービス - プリンシパル、グループ、オーガニゼーション、アカウント、および管理ポリシーの複製されたデータベースである、レジストリー・データベースを保守します。
- 認証サービス - ユーザーの認証、あるいはプリンシパルが正しく指定されていることを検証するプロセスを取り扱います。認証サービスは、プリンシパルがリモート・サービスにアクセスするために使用するチケットも発行します。チケットには、サービスを要求する側のプリンシパルがサービスを提供する側のプリンシパルへ提示するデータが含まれています。
- 特権サービス - 要求された操作を実行する権限をプリンシパルが確実に保持しているようにするための、ユーザーの特権属性を提供します。

そのほかに、DCE セキュリティー・サービスは以下のものを提供します。

- アクセス制御リスト (ACL) 機能 - オブジェクトのアクセス認可に基づいてオブジェクトへのアクセス権を確立し、与えます。
- 拡張レジストリー属性 (ERA) 機能 - レジストリー・データベース・スキーマを拡張して追加属性を定義するツール、およびこれらの属性をレジストリー・オブジェクトに付加するツールを提供します。

DCE ホスト・デーモン (**dced**) は、セキュリティー・クライアントの働きをします。

DCE のレジストリー、認証、および特権の各サービスは、単一のデーモンであるセキュリティー・サーバー (**secd**) として実装されています。

DCE 認証サービス・サーバーおよびクライアント

認証サービスは、レジストリー・データベース、セキュリティー・サーバー、およびセキュリティー・クライアントで構成されています。セキュリティー・クライアントは、セキュリティー・サーバー (*dcelocal*/**bin/secd**) と通信を行い、情報や操作を要求します。セキュリティー・サーバーは、レジストリー・データベースにアクセスして、照会と更新を行い、またユーザーのログインを検証します。レジストリー・データベースにアクセスするために、認証サービスはレジストリー・サービスと対話する必要があります。 318ページの図44 は、セキュリティー・クライアント、サーバー、およびレジストリー・データベース間の関係を単純化して表したものです。

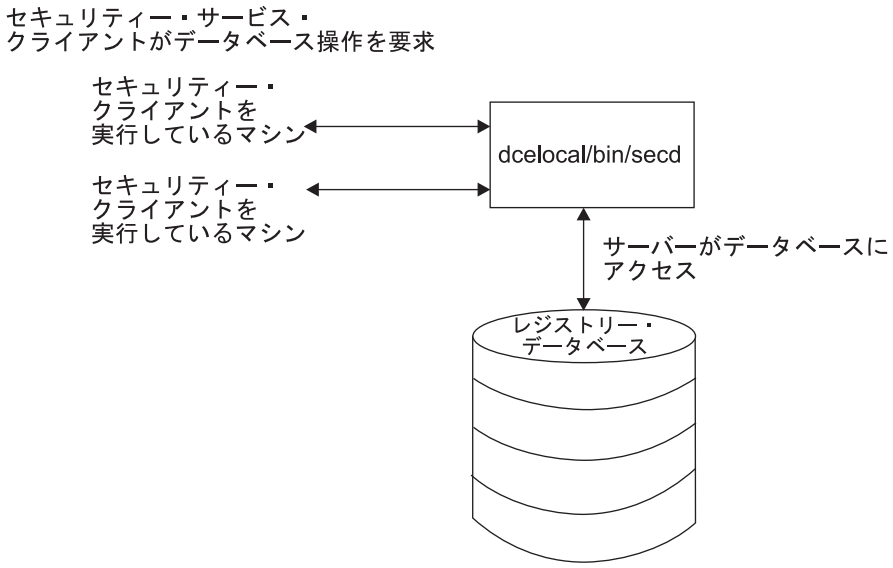


図 44. マシン、サーバー、およびデータベース

推奨セキュリティー・サーバー・レプリカ

DCE の中では、複製されたデータ (レプリカ) のクライアントによる使用を、優先順位付けすることが望ましい場合があります。セキュリティー・サーバーのレプリカの場合、これを行うには、各クライアント上に **/opt/dcelocal/etc/security/pe_site** ファイルを作成して望ましい優先順位を反映し、各クライアントがこのファイルを使用してセル内のセキュリティー・サーバーを見つけるように構成します。この作業を管理できるようにするために、 DCE for AIX および Solaris バージョン 3.2 ではいくつかの機能強化が行われています。

1. **setprofile.tcl** というスクリプトが DCE クライアントに付属しています。セル管理者はこのスクリプトを使用して、セル内の各セキュリティー・サーバーに優先順位を付けることができます。推奨レプリカの設計は、 RPC プロファイルを使ってこの情報を格納することを前提としています。特に、このスクリプトは指定されたクライアントの LAN プロファイルを変更します。

   このスクリプトを実行するには、セル内のいずれかの完全なクライアントから **DCE cell_admin** としてログインし、 AIXまたはSolaris コマンド行で、**setprofile.tcl** という対話式のスタンドアロン推奨レプリカ・スクリプトを実行します。

   唯一の引き数として、更新するクライアントの DCE ホスト名が必要です。このスクリプトは、使用可能なセキュリティー・サーバー・レプリカのリストを表示し、対話式で優先順位を設定できます。可能な優先順位は 0 から 7 であり、0 がレプリカの最高優先順位です。

   注: このスクリプトを実行する必要があるのは、セル内で使用される各 LAN プロファイルごとに一度だけです。

構成中に、それぞれの DCE クライアントに対して、 LAN プロファイルへのポインターが 1 つ割り当てられます。特定のクライアントにどのプロファイルが構成されたかを見つけるには、次の dcecp コマンドを実行します。

```
rpcprofile list/.:/hosts/{dce_hostname}/profile
```

ここで **{dce_hostname}** は、クライアントを DCE セル内に構成するために使用された名前です。 AIX および Solaris のデフォルト LAN プロファイルは、**/.:/lan-profile** です。

クライアントのグループが、指定されたセキュリティー・サーバー設定変更のセットを使用できるようにするには、デフォルトの LAN プロファイルを変更します。たとえば、地理上の場所 A のクライアント・グループは、場所 A のセキュリティー・サーバーへ優先順位を与える、 lan-profileA というプロファイルを使用し、別の場所 B のクライアント・グループは、場所 B の近くのセキュリティー・サーバーへ優先順位を与える、 lan-profileB というプロファイルを使用することが望ましい場合があります。

LAN プロファイルからレプリカの優先順位を削除するには、 **setprofile.tcl** スクリプトを再度実行するか、または **dcecp rpcprofile remove** コマンドを使用します。たとえば、 rep1 というセキュリティー・レプリカを lan-profile から削除するには、次のように入力します。

```
rpcprofile remove -i d46113d0-a848-11cb-b863-08001e046aa5,2.0
                  -m /.:/subsys/dce/sec/rep1
                   /.:/lan-profile
```

この例で、**d46113d0-a848-11cb-b863-08001e046aa5,2.0** は、セキュリティー・サーバーのインターフェース ID です。

2. DCE ホスト・デーモン dced は、 **pe_site** ファイルを 24 時間ごとに自動的に再作成します。セキュリティー・サーバー設定変更が確立されている場合、この設定変更を反映するために、 dced はセル内のそれぞれの完全クライアント上に **pe_site** ファイルを再作成します。クライアント上の **pe_site** ファイルを即座に更新した方が望ましい場合には、 **dcecp secval update** コマンドを実行します。
3. **TRY_PE_SITE** 環境変数は、クライアント構成プログラムによってデフォルトにセットされます。この環境変数がセットされている場合、 DCE クライアント上で実行するプロセスは、 pe_site ファイル内の情報を使用してセル内のセキュリティー・サーバーを探します。 **pe_site** ファイル内にリストされているレプリカのいずれともコンタクトがとれない場合、クライアントは、 **/.:/sec** 内の DCE ディレクトリー・サービス内に保持されているセキュリティー・サーバーのリストを使用して、レプリカを探そうとします。

レジストリー・データベース

レジストリー・データベースには、次の情報が含まれています。

- プリンシパル - プリンシパルは、システムのユーザーです。プリンシパルは、対話式プリンシパル (ユーザー)、または非対話式プリンシパル (サーバー、マシン、およびセル) です。プリンシパルは、アクセス許可に関連付けることができます。

- グループ - グループは、グループ名によって識別される、プリンシパルの集合です。グループをアクセス許可に関連付けることができます。
- オーガニゼーション - オーガニゼーションは、オーガニゼーション名によって識別されるプリンシパルの集合です。オーガニゼーションは、レジストリー内のプリンシパルに関連したポリシーを定義します。オーガニゼーションは、アクセス許可に関連付けられません。
- アカウント - アカウントは、プリンシパルがセル内のオブジェクトへの認証されたアクセスを行えるようにする、パスワードとアカウント情報を含んでいます。(認証されたアクセスは、以下で説明するように、異なるセルのプリンシパルの間でも行われます。)
- ポリシーおよびプロパティー - ポリシーとプロパティーは、パスワードの長さや形式、およびある種の認証要件などを規制します。
- **replist** オブジェクト - このオブジェクトは、レジストリー・データベースのレプリカを管理するために使用されます。
- **xattrschema** オブジェクト - このオブジェクトは、 ERA 機能によって作成された拡張レジストリー・スキーマです。

(レジストリー・データベースの構造、およびそこに含まれている情報のタイプについての詳細は、 533ページの『第43章 レジストリー・オブジェクトへのアクセス』を参照してください。)

レジストリー・データベースによって管理されるオブジェクトの集合が、*セル* と呼ばれるエンティティーです。 セル間の認証された通信は、それぞれ通信相手のセルのレジストリー・データベースにそのセルの特別のアカウントがある場合にのみ可能です。これらの特別のアカウントは、あるセルのプリンシパルから他のセルへの認証されたアクセスを可能にする、セル間認証を設定します。 (セル間認証用のアカウント設定については、 431ページの『第33章 マルチセル環境の管理』を参照してください。)

データベースの物理的セキュリティー

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

DCE セキュリティー・サービスは、ネットワーク・セキュリティーのための保護手段を提供し、コンピューター間で通信するプリンシパルの識別を保証することにより、ネットワークを超えて伝送される情報を保護します。ただし、セキュリティー・サーバーとそのデータベースは、ローカル・マシンに常駐します。レジストリー・データベースは、それが常駐するマシンに実装されたセキュリティーと同程度にしか安全ではありません。ローカル・マシン上の機密データにはルートしかアクセスできないようにするとともに、セキュリティー・サーバーが常駐するマシンの物理的セキュリティーを確立する必要もあります。そのために、マシンを鍵のかかる部屋に設置したり、マシンを使用した時刻とユーザーについて利用記録を取ったりするなど、必要に応じた措置を講じてください。

注: 認証についての詳細は、*IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* を参照してください。

レジストリー・データベースの格納方法

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

各セキュリティー・サーバーは、仮想記憶にレジストリー・データベースの作業用コピーを、またディスクに永続コピーをそれぞれ保持します。読み取りと更新はすべて仮想記憶内のコピーに対して行われます。サーバーは、始動時にディスク上のコピーを使用して、仮想記憶内のコピーを初期設定します。サーバーに障害があった場合にデータベースの状態を保証するために、自動更新ログが使用されます。

図45 は、サーバーおよびレジストリー・データベースのディスク記憶コピーと仮想記憶コピーを示しています。

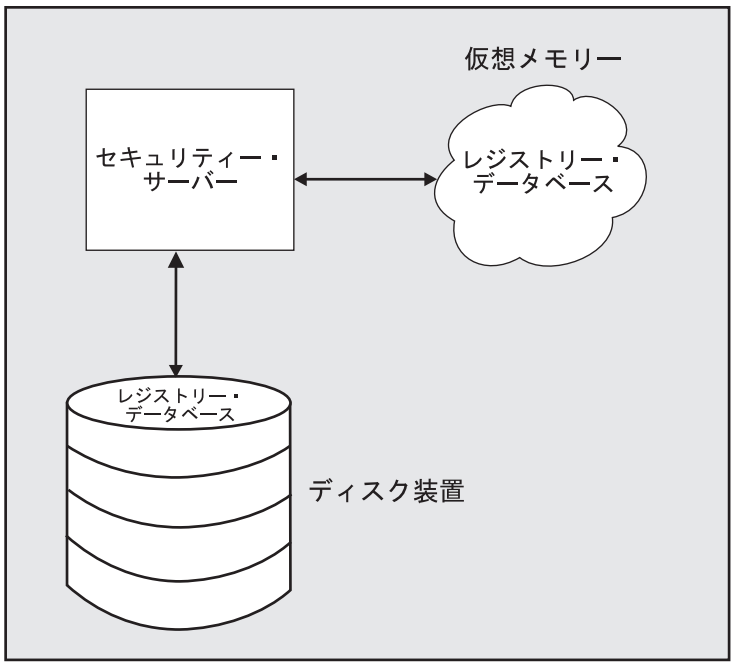


図 45. レジストリー・データベースのディスク記憶コピーおよび仮想記憶コピー

各セキュリティー・サーバーは、定期的に仮想記憶からディスクへデータベース全体を保管します。データベースは、*dcelocal*/**var/security/rgy_data** に保管されます。

複製データベース

レジストリー・データベースは、セル内で複製することができます。セルのセキュリティー・サーバーの各インスタンスは、データベースの作業用コピーを保持します。本書では、セキュリティー・サーバーとそのデータ (レジストリー・データベース) の組み合わせを、*レプリカ* と呼びます。通常、パフォーマンスと信頼性を高めるために、セル内にいくつかのレプリカを作成します。

セルのレプリカの整合性を保つ作業は、セキュリティー・サーバーによって自動的に行われます。すべての変更は、以下のセクションで説明するように、すべてのレプリカに自動的に反映されます。

更新処理方法

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

更新は 1 つのデータベースに対してだけ行われ、その変更内容は他のデータベースに伝えられます。この伝搬が保留されている間、レプリカが完全に最新の状態ではなくても、レプリカへのアクセスは可能です。つまり、変更がまだ適用されていないレプリカも使用可能です。このような複製メカニズムにより、変更が伝搬処理中であっても、すべてのレプリカをログイン妥当性検査や読み取り操作に利用することができます。

マスター・レプリカとスレーブ・レプリカ

マスター・レプリカ というセル内のただ 1 つのレプリカだけが、クライアントからのデータベース更新を受け入れます。他のレプリカは *スレーブ・レプリカ* と呼ばれ、クライアントからの読み取りのみを受け入れます。マスター・レプリカは、更新をスレーブ・レプリカに伝搬します。たとえば、マスター・レプリカとスレーブ・レプリカはいずれも、 **/bin/login** のようなアカウント情報をクライアント・プログラムに提供できます。しかし、アカウントの追加やパスワード情報の変更の場合は、それらの更新をマスター・レプリカでのみ処理できます。

データベースの更新処理は、マスター・レプリカとスレーブ・レプリカでは多少異なります。 323ページの図46 と 324ページの図47 は、マスターとスレーブの更新処理を図示しています。図の後のセクションで各プロセスについて説明します。

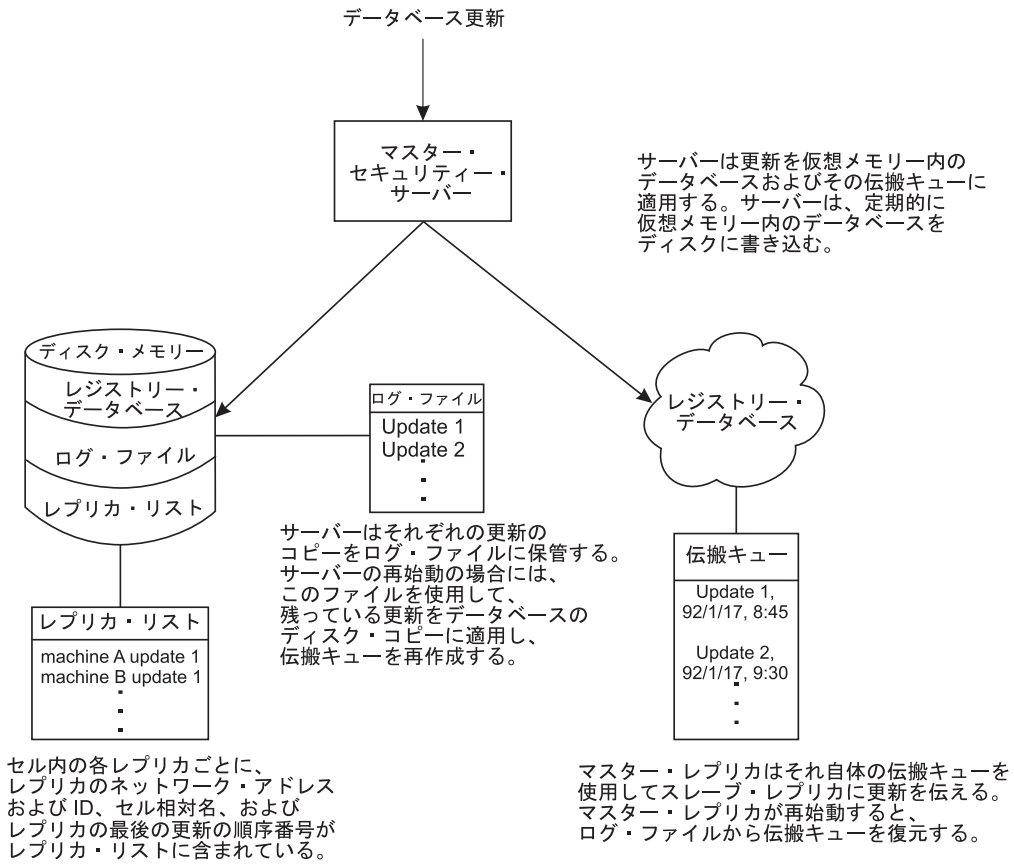


図 46. マスター・レプリカ更新処理

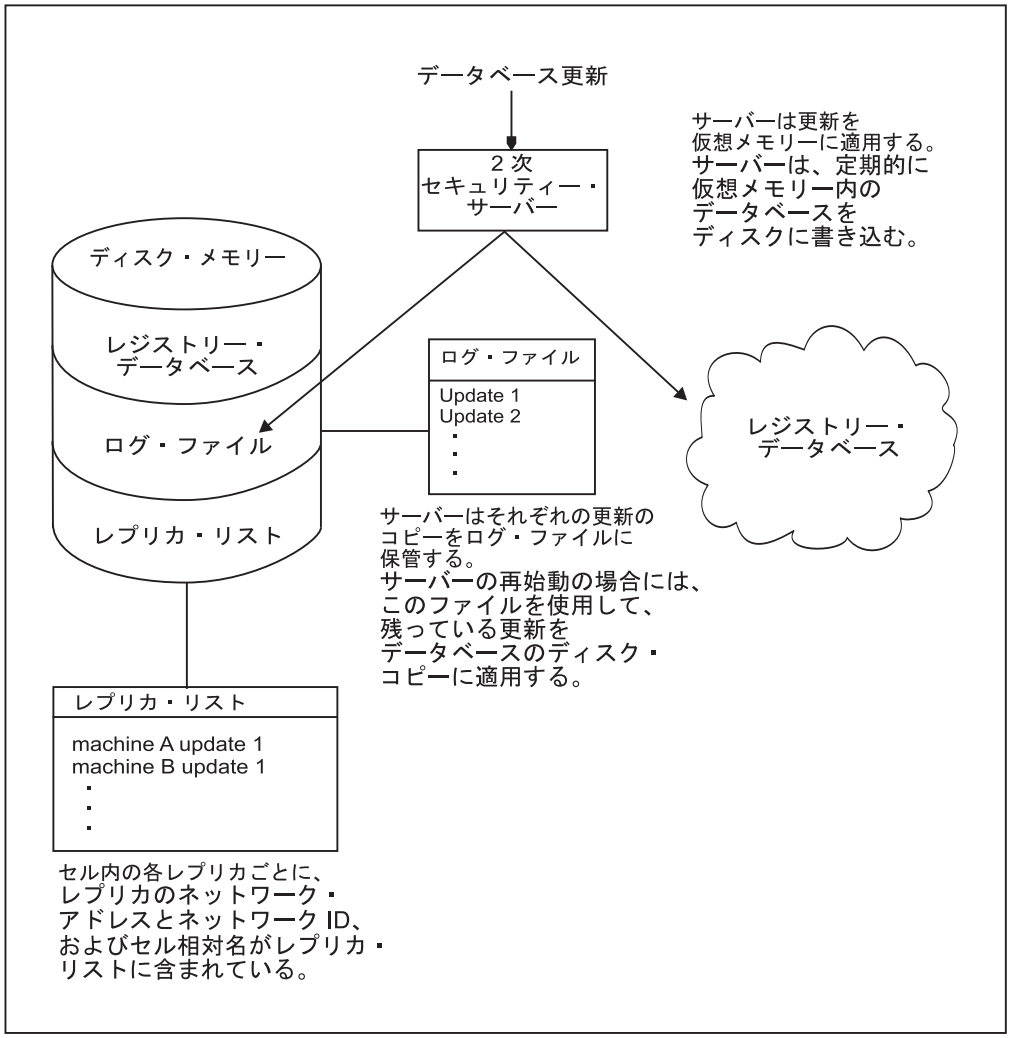


図 47. スレーブ・レプリカ更新処理

データベース更新の処理

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

マスター・レプリカあるいはスレーブ・レプリカは、更新を受け取ると仮想記憶内のデータベースに更新を適用し、ディスクに格納されているログ・ファイルにそれぞれの更新のコピーを保管します。更新は、ログ・ファイルの中に番号順に累積されます。サーバーが予期せずに再スタートした場合でも、ログ・ファイルがあるため、更新が失われることはありません。

レプリカは定期的に仮想記憶内のデータベースをディスクに書き出して、ディスク・コピーを最新の状態に保ちます。その後、レプリカがスレーブの場合は、すべての更新のログ・ファイルを消去します。マスターの場合は、スレーブ・レプリカ

にすでに伝搬された更新のログ・ファイルをすべて消去します。スレーブに伝搬されていない更新は保存され、必要に応じて、伝搬キューの再構築に使用されます。

マスター・レプリカだけが伝搬キューを保持します。これを使用して、スレーブ・レプリカに伝搬すべき変更を保管します (これについては 『データベース変更の伝搬』で説明しています)。マスター・レプリカは更新を受け取ると、それを仮想記憶データベースとログ・ファイルのほかに、伝搬キューにも追加します。伝搬キュー内の各更新は、順序番号とタイム・スタンプによって識別されます。順序番号は、スレーブ・レプリカへの更新の伝搬を追跡するために内部で使用されます。タイム・スタンプは、更新の日時をユーザーに示します。

マスター・レプリカまたはスレーブ・レプリカが再スタートするとき、仮想記憶のデータベースを初期設定し、次にログ・ファイル内の未解決の更新をデータベースに適用します。レプリカがマスター・レプリカの場合は、レプリカはログ・ファイルから伝搬キューを再作成し、未解決のスレーブ更新があれば、それが伝搬されるようにします。このメカニズムにより、サーバーがシャットダウンされたときでも、更新が失われることはありません。

データベース変更の伝搬

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

スレーブ・レプリカに更新を伝搬するために、マスター・レプリカは 324ページの『データベース更新の処理』に説明されているプロセスによって、まずデータベースのコピーを更新します。次にマスター・レプリカは、レプリカ・リストにあるそれぞれのスレーブ・レプリカに更新を伝搬します。レプリカ・リストには、各スレーブ・レプリカの ID とネットワーク・アドレスが含まれています。また、このリストには、スレーブに対して最後に行われた更新の順序番号も入っています。マスター・レプリカは、常に順序番号の順に伝搬します。レプリカ・リストのレプリカに関連付けられた順序番号と、伝搬キューにある更新の順序番号を調べることにより、マスターは伝搬キュー内のどの更新をどのスレーブに伝搬すべきかを判断できます。このメカニズムにより、1 つのスレーブ・レプリカが使用不能になっても、残りのスレーブ・レプリカの更新に影響することはありません。

更新の伝搬が最初の試行で成功しない場合、マスター・レプリカは、成功するまで定期的に試みます。更新が正常に行われると、マスターは、レプリカ・リスト上の更新されたレプリカに関連付けられた順序番号を更新します。更新がすべてのスレーブ・レプリカに伝搬されると、マスター・レプリカは伝搬キューから更新を除去します。

マスター / スレーブ認証

すべての DCE オブジェクトと同様に、マスター・レプリカとスレーブ・レプリカは、互いに認証する必要があります。これを行うため、マスターはデータベース作成時に作成されるプリンシパルの 1 つである、 **dce-rgy** の識別を使用します。スレーブは、それが存在するホスト・マシンの識別を持っています。この識別は、

/.:/sec/replist オブジェクト (533ページの『第43章 レジストリー・オブジェクトへのアクセス』を参照) への権限を持っていなければならないことに注意してください。

ユーザー間認証

ユーザー間認証機能は、保護 RPC を受信する必要があるものの、長期間使用されるキー (キー・テーブル・ファイルなど) にアクセスできないアプリケーションで使用するためのものです。

この機能は、メッセージング・モデルを使用するクライアント / サーバー・アプリケーションでは必須です。このモデルを使用して、クライアントは何らかのアクティビティーを要求するためにサーバーへの RPC を行います。サーバーは要求をキューに入れ、即時にクライアントに戻ります。次いでクライアントは、サーバーがその要求を完了してコールバックするのを待ちます。このシナリオでは、サーバーはコールバックのための保護 RPC を開始し、クライアントはサーバーの役割を果たします。

ファイル /etc/passwd と /etc/group およびレジストリー

UNIX プログラムとの互換性を維持するために、ローカル・マシン上に **/etc/passwd** ファイルと **/etc/group** ファイルの標準バージョンを保持する必要があります。 **/etc/passwd** ファイルおよび **/etc/group** ファイルのレジストリー・データベースとの整合性を保つために、 **passwd_export** コマンドを使用します。 (できれば **cron** を使用して) **passwd_export** を定期的に実行することを推奨します。 (**passwd_export** の詳細については、 457ページの『第36章 ルーチン保守の実行』を参照してください。)

注: 標準の UNIX の動作と異なり、 **/etc/passwd** および **/etc/group** ファイルは、セキュリティー・サーバーが使用できない場合は、ローカル・ログインに対して使用されません。代わりに、ローカル・レジストリー (次のセクションを参照) が使われます。 **/etc/passwd** および **/etc/group** ファイルは、これらの必要とする UNIX プログラムとの互換性のためにのみ 保持されています。

ローカル・レジストリー

各ローカル・マシン上の *dcelocal***/var/security** ディレクトリーにあるローカル・レジストリーは、マシンの最新のユーザーと、ユーザーが最後にログインした日時についての情報を含んでいます。ネットワーク・ログインでセキュリティー・サーバーが使用できない場合、認証サービスは、ローカル・ログインに必要な情報をローカル・レジストリーから取得することを試みます。

セキュリティー・サーバーがネットワークで実行されている場合、任意のユーザーがマシンから DCE に初めてログインしたときに、認証サービスはローカル・レジストリーを自動的に作成します。その後、任意のユーザーがマシンから DCE にログインするたびに、ローカル・レジストリーが更新されます。ローカル・レジストリーを編集するには、 **-l** フラグを指定した **rgy_edit** コマンドを使用します。 **dcecp** は、ローカル・レジストリーにアクセスしないことに注意してください。

セキュリティー・オブジェクトの名前

セキュリティー・ネームスペースのルートはセル・ディレクトリー・サービス (CDS) ネームスペースにあるため、セキュリティー・オブジェクトの CDS パス名は次のようになります。

/.../cellname/mount_point/object_name

ここで、

cellname
オブジェクトが存在するセルの名前。

mount_point
セキュリティー・サービスが CDS に登録される名前。

object_name
オブジェクトが作成されるときに割り当てられる、レジストリー・オブジェクトの名前。オブジェクトがディレクトリー内にある場合、*object_name* は、オブジェクト自体の名前、およびオブジェクトへのアクセスのために走査するすべてのディレクトリー名で構成されます。一般に、レジストリー・オブジェクトは、レジストリー・データベース内のプリンシパル、グループ、またはオーガニゼーションのディレクトリーに存在することに注意してください。レジストリー・データベースの構造の詳細については、 533ページの『第43章 レジストリー・オブジェクトへのアクセス』を参照してください。

たとえば、プリンシパル **bach** がセル **dresden.com** に存在し、 **sec** (セキュリティー) マウント・ポイントを使用し、 **principal** ディレクトリーにある場合、このプリンシパルの全パス名は以下のようになります。

```
/.../dresden.com/sec/principal/bach
```

別の例として、グループ **east-west** が、セル **dresden.com** のレジストリー・データベース内のディレクトリー **group** のサブディレクトリー **sales** に存在すると想定します。この場合、**east-west** の全パス名は次のようになります。

```
/.../dresden.com/sec/group/sales/east-west
```

dcecp セキュリティー・コマンドでの名前の使用

DCE セキュリティー・サービスの管理のために使用するすべての **dcecp** コマンド (ただし **dcecp acl** を除く) では、操作したいオブジェクトを識別するオブジェクト名のみを指定します。オブジェクト名は、レジストリー・データベースに収められています。セル名 (ローカル・セルが想定される) やマウント・ポイント (セキュリティー・サービスに登録された名前が想定される) を入力する必要はありません。

dcecp acl コマンドでの名前の使用

他の **dcecp** セキュリティー・コマンドと異なり、 **dcecp acl** コマンドは、セキュリティー以外の DCE サービスが保守できる ACL に対して働きます。 (異なるネ

ームスペースに分かれて存在することのあるオブジェクトを操作する) 汎用ツールと同様に、 **dcecp acl** は、オブジェクトの単なる *object_name* ではなく、完全修飾 CDS パス名を必要とします。

たとえば、プリンシパル **bach** のレジストリー・アカウントに関係する ACL を変更するために **dcecp acl** コマンドを使用する場合、次に示す完全修飾名を入力する必要があります。

```
/.../dresden.com/sec/principal/bach
```

または

```
/.:/sec/principal/bach
```

また、(プリンシパルを追加あるいは削除できる担当者を管理するために) **dcecp acl** を使ってレジストリー・データベースのプリンシパル・ディレクトリーにある ACL を処理するには、次に示す完全修飾名を入力する必要があります。

```
/.../dresden.com/sec/principal
```

このセル名は、次のようなディレクトリーの名前も表せます。

```
/.:/sec/principal/vienna.com/violinists_cell
```

これらの場合、**dcecp acl** コマンドには、入力しているのがディレクトリー名またはプリンシパル名のどちらであるかを識別するオプションがあります。

第28章 アクセス制御リストの使用

DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

DCE オブジェクトへのアクセスは、ACL 許可メカニズムを使用して制御できます。 ACL は、ファイル、ディレクトリー、CDS エントリー、およびレジストリー・オブジェクトに関連付けられます。また、任意のアプリケーションは、自分の内部データ・オブジェクトへのアクセスを制御するために ACL を実装することができます。各 ACL は、オブジェクトに対して許可を与えられるユーザーおよび許可の内容を定義する、複数の ACL エントリーで構成されています。それらのエントリーは以下のとおりです。

- オブジェクトにアクセスできるユーザー
- それらのプリンシパルまたはグループがオブジェクトに対して許可されている、アクセスの種類
- 認証されていないユーザーに許可されるアクセスの種類

特定の DCE コンポーネントが ACL 認可メカニズムを実装する方法について、詳しくは本書の該当するトピックを参照してください。

注: この章およびこれ以降の章での DCE 認可の説明において、ユーザー という用語はプリンシパルに類似しています。プリンシパルは、人間のユーザー、サーバー、あるいはマシンのいずれかです。

認可の概要

ACL には、オブジェクトにアクセスできるプリンシパル、およびこれらのプリンシパルが実行できる操作を指定するエントリーのリストが格納されています。プリンシパルは、明示的に指定することができます。また、ACL エントリー内で識別されるグループのメンバーでもかまいません。 ACL は、それが保護するオブジェクトに関連付けられています。プリンシパルが実行できる操作は、許可によって指定されます。

DCE 許可は、以下のものに対して設定することができます。

- 所有者、グループ、およびその他
- ローカル・セルおよび外部セル内の特定の個別のプリンシパル
- ローカル・セルおよび外部セル内の特定の個別のグループ
- 特定の外部セル内にある、個別の許可が設定されていない他のすべてのプリンシパル
- DCE 認証サービスによって認証された、すべてのセル内のすべてのプリンシパル
- ローカル・セルあるいは外部セル内の代行ユーザー、サーバー、あるいはグループ
- 許可されていないユーザー

ACL は、マスキング機能、および現行のバージョンとは異なる DCE バージョンからの保護を組み込むための方式も提供します。

ファイル・システムは、多くの場合、ファイル・システムのオブジェクト (ファイルやディレクトリーなど) へのアクセス許可を提供するように設計されています。 DCE の ACL はより拡張されています。 DCE では、多くのオブジェクトが ACL を持つことができ、また多くのオブジェクトに許可を割り当てることができます。 DCE ACL は、分散ファイル・サービス、DCE セキュリティー・サービス、 DCE ディレクトリー・サービスなどの、 DCE コンポーネントによって管理されるオブジェクトへのアクセスを制御します。

たとえば、セキュリティー・サービス (アカウントを管理するコンポーネント) の ACL は、一部のプリンシパルに対してアカウントに関するすべての情報の変更を許可し、別のプリンシパルに対してアカウントに関する情報のサブセットのみの変更を許可し、さらに別のプリンシパルに対してアカウントに関する情報の変更を禁止することができます。

DCE は、特定のオブジェクトのタイプに対応する、特定の許可のセットをサポートします。たとえば、コンテナーの場合、他のオブジェクト (プリンシパルなど) には必要のない「挿入」許可があります。 ACL のこの広範な使用は、たとえば、 POSIX システムとは対照的です。 POSIX では標準の許可セット (読み取り、書き込み、および実行) を使って、ファイル・システム・オブジェクトのみを許可ビットによって保護します。 DCE 制御プログラムには **acl permissions** というコマンドがあり、指定されたオブジェクトに関連付けられた ACL 特有の許可を表示します。

ACL マネージャー

ACL マネージャーは、ACL を処理するサーバーの一部です。 1 つの ACL マネージャーは数種類の ACL をサポートできます。より抽象的な観点から言うと、それぞれの ACL タイプは、対応する ACL マネージャー・タイプによってサポートされます。 ACL マネージャー・タイプは非公式に ACL マネージャーと呼ばれることもあります。 331ページの図48 は、サーバー内の ACL マネージャーを示しています。

クライアント側では、 ACL インターフェースをエクスポートする任意のサーバーに接続できるため、 1 つのプログラムですべての ACL を操作できます。 DCE 制御プログラムはこの機能を使用します。

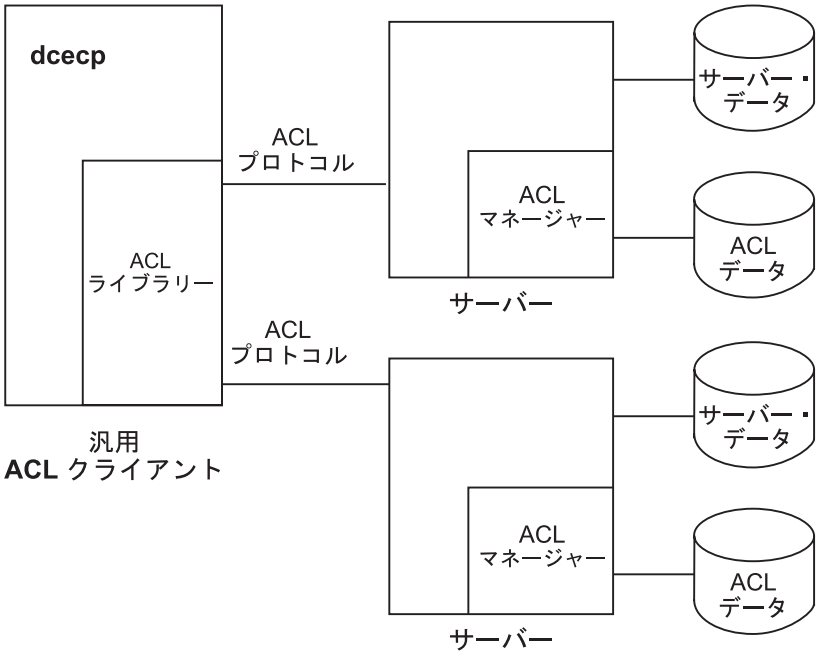


図 48. サーバーの ACL マネージャー

標準の DCE コンポーネントに加えて、ACL は、 ACL マネージャーが実装されているオブジェクトへのアクセスを制御することができます。 ACL をユーザー作成のアプリケーションに関連付けて、アプリケーション自体、アプリケーション内のファイル、およびそれらのファイルのフィールドへのアクセスを保護することができます。

ここで説明する ACL の要素はすべて ACL マネージャーで使用できます。ただし、各マネージャーはすべての要素を実装している場合もあり、要素のサブセットだけを実装している場合もあります。特定の DCE コンポーネントが ACL を使用する方法については、本書の該当するセクションを参照してください。

OSF DCE リリース 1.0.3 で提供されるコード例を使用して ACL を構築してある場合は、 DCE の新しいバージョンでアプリケーションを実行するには、 daclmgr.h ファイルと、ACL マネージャーを使用するアプリケーションを変更する必要があります。

1. daclmgr.h を変更します。 API プロトタイプ (**sec_acl_mgr_get_access** および **sec_acl_mgr_is_authorized**) の中のデータ・タイプ `rpc_authz_cred_handle_t` を `sec_id_pac_t` に置換します。

```
extern void DCEAPI sec_acl_mgr_get_access(
#ifdf IDL_PROTOTYPES
/* in */ sec_acl_mgr_handle_t sec_acl_mgr,
/* in */ rpc_authz_cred_handle_t *accessor_info, <-This line becomes
/* in */ sec_id_pac_t *accessor_info, <-this line.
/* in */ sec_acl_key_t_sec_acl_key,
/* in */ uuid_t *manager_type,
/* in */ sec_id_t *user_obj,
/* in */ sec_id_t *group_obj,
/* out */ sec_acl_permset_t *net_rights,
#endif
extern boolean32 DCEAPI sec_acl_mgr_is_authorized(
#ifdef IDL_PROTOTYPES
```

```
/* in */ sec_acl_mgr_handle_t sec_acl_mgr,
/* in */ sec_acl_permset_t desired_access,
/* in */ rpc_authz_cred_handle_t *accessor_info, <-This line becomes
/* in */ sec_id_pac_t *accessor_info, <--this line.
/* in */ sec_acl_key_t sec_acl_key,
/* in */ uuid_t *manager_type,
/* in */ sec_id_t *user_obj,
/* in */ sec_id_t *group_obj,
/* out */ error_status_t *st
#endif
```

2. アプリケーションで、変数 rdaclif_v0_0_epv_t を rdaclif_v1_0_epv_t に変更します。
3. アプリケーションで、変数 rdaclif_v0_0_s_ifspec を rdaclif_v1_0_s_ifspec に変更します。

ACL の解釈

アカウントに関連付けられた情報には、プリンシパルおよび一連のグループが含まれます。 (この場合、グループは、その起源が Multics にあることから、*プロジェクト・リスト* と呼びます。) プリンシパルとプロジェクト・リストを一緒にして、アカウントに関連する*特権属性* (またはクライアント側アクセス制御情報) と呼びます。

プリンシパルとそれぞれのグループは、ストリング名と UUID の両方によって表されます。特権属性 UUID は、認証されたリモート・プロシージャー・コール (RPC) または GSSAPI コールで使われる証明書の中に含まれています。サーバーは、受け取った証明書の内容に基づいて、アクセスを与えます。通常、サーバーは認証されていないコール (証明書をもたないコール) を拒否しますが、それらを受け入れるというポリシーをいずれのサーバーもサポートできます。その場合、そのような認証されていないクライアントに与えられるアクセスをサーバーが追加的に制限できるようにするために、サーバーの ACL マネージャーが **unauthenticated** マスク ACL エントリー・タイプをサポートする必要があります。

プリンシパルが ACL に関連した DCE オブジェクトへのアクセスを要求すると、オブジェクトの ACL マネージャーは、プリンシパルとそのプリンシパルがメンバーである、グループの UUID (プリンシパルの特権属性) と、 ACL エントリーにリストされているプリンシパルとグループの UUID とを比較します。これは、単に ACL エントリーのリストを読むことによって行われます。マネージャーは、プリンシパルのいずれかの特権属性と一致する、最初に検出した ACL エントリー (あるいはグループの場合は複数のエントリー) 内のアクセス認可を与えます。一致するエントリー内の許可によって、要求されたアクセス・モードが許される場合、プリンシパルにアクセスが与えられます。そうでなければ、アクセスは拒否されます。

プロセスによって継承される証明書

プリンシパルによって作成、つまり生成されたプロセスは、プリンシパルの証明書を継承します。たとえば、ユーザーがログインし、認証され、アプリケーションを開始すると、開始されたアプリケーションは、ユーザーの認証された証明書を継承して、あたかもユーザーであるかのように動作します。特定のオブジェクトに対するアプリケーションの許可は、ユーザーの許可と同じです。アプリケーションによって生成されたプロセスは、ユーザーの識別を受け継ぎ、さらにそれらのプロセスが開始するプロセスに識別を渡します。

注: **setuid** 許可ビットの変更は、ネットワーク識別ではなく、実行可能ファイルが動作しているローカル・オペレーティング・システムの識別だけを変更します。

サーバーの中には、個別の認証されたプリンシパルとして動作するように書かれているものがあります。これらのサーバーについては、システム管理者はレジストリー・データベースにアカウントを作成します。これらのサーバーを開始すると、サーバー・プロセスはレジストリーを使って認証を行い、証明書を受け取り、ユーザーの識別ではなく、サーバー・プロセス自身の識別のもとで動作します。

ACL エントリーおよびマスク

ACL エントリーには、いくつかの異なる *ACL エントリー・タイプ* があり、各タイプには特定の目的があります。すべての ACL エントリーは同じ形式のリスト構文で表されます。

ACL 構文

DCE 制御プログラムは、Tcl 言語がサポートするコマンド構文を使用します。 Tcl では、ACL エントリーを表すリストには、 ACL エントリーのタイプに応じて 2 つまたは 3 つの要素が含まれ、次のような形式になります。

{*type* [*key*] *permissions*}

図49 の 3 つのサンプル ACL エントリーは、 Tcl が入力として受け入れるフォーマットです。

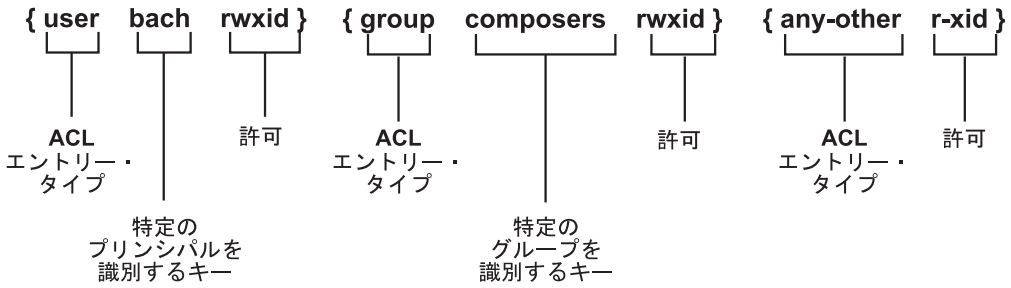


図 49. サンプル ACL エントリー

最初のサンプル ACL エントリーは、ローカル・セル **bach** 内のあるプリンシパルのための許可をセットします。 ACL エントリー・タイプは **user**、キーは **bach**、許可は **rwxid** です。エントリーのコンポーネントは、スペース文字で区切られています。

2 番目のサンプル ACL エントリーは、ローカル・セル **composers** 内のあるグループのための許可をセットします。 ACL エントリー・タイプは **group**、キーは **composers**、許可は **rwxid** です。

3 番目のサンプル ACL エントリーは、 (より特定化されたエントリーに一致しない) ローカル・セルまたは外部セル内の他のすべてのプリンシパルのための許可をセットします。 ACL エントリー・タイプは **any-other**、キーはなし、許可は **r-xid** です。必ずしも ACL のすべてのタイプにキーが必要であるとは限りません。

出力として、ACL 許可の Tcl フォーマットに、それぞれの許可に対する許可文字または - (ダッシュ) のいずれかが含まれます。次に 2 つの例を示します。

```
{user mozart crwx---}
{user brahms -------}
```

入力として、出力フォーマットと同じものを使用できます。また、ダッシュを省いたゆるやかな形式を使用することもできます。入力の場合、上と同じ例を以下のように短縮することができます。

```
{user mozart crwx}
{user brahms -}
```

ユーザー **brahms** がすべての許可を与えられていないことを示すために、ダッシュを 1 つだけ使用します。

プリンシパルおよびグループのための ACL エントリー・タイプ

ACL エントリー・タイプによって、次のもののエントリーを定義することができます。

- プリンシパルおよびグループ
  - ローカル・セル内のプリンシパルおよびグループ
  - 外部セル内のプリンシパルおよびグループ
  - 代行エントリー
  - 個々の ACL エントリーが作成されていない、ローカル・セル内のすべてのプリンシパル
  - 特権属性が他の ACL エントリーのいずれとも一致しない、ローカル・セルおよび外部セル内のすべてのプリンシパル
- 認証されたユーザーおよび認証されていないユーザーに使用されるマスク
- 異なる DCE リリースによるコピーおよび表示 (解釈されない場合) が可能な、今のところ未定義になっているエントリー・タイプ

プリンシパルまたはグループが認証されていない場合、エントリー内の許可は、**unauthenticated** マスク (これについてはこの章の後の部分で説明) によって追加的に制限されます。認証されたプリンシパルのエントリーは、 **user_obj** エントリーおよび **other_obj** エントリーを除いてすべて、 **mask_obj** マスク (これについてもこの章の後の部分で説明) によって追加的に制限されます。

次のリストは、プリンシパルとグループのエントリー・タイプ、それらの意味、およびそれらの入力フォーマットをまとめたものです。すべての ACL には、表で参照しているように、デフォルト・セルが定義されています。これは変更可能であり、さまざまなデータ・タイプ用にセルを定義する場合に使用できます。

このリストは、以下の構文変数を使用しています。

principal_name
: レジストリー・データベース内のプリンシパルの名前。

group_name
: レジストリー・データベース内で定義されているグループの名前。

cell
: セルのグローバル・パス名、フォーマットは */.../name*。

permissions
オブジェクトの ACL マネージャーによって使用可能にされる許可。

プリンシパルとグループの ACL エントリー・タイプは以下のとおりです。

user_obj
オブジェクトの実際のユーザーあるいは実効ユーザーの許可を確立します。たとえば、ファイルの所有者です。入力フォーマットは次のとおりです。

{**user_obj** *permissions*}

group_obj
オブジェクトの実際のグループあるいは実効グループのメンバーの許可を確立します。たとえば、ファイルのグループです。入力フォーマットは次のとおりです。

{**group_obj** *permissions*}

other_obj
デフォルト・セル内の他のすべてのプリンシパルのための許可を確立します。ただし、プリンシパルがエントリー・タイプ **user** として ACL で指名されていない場合、エントリー・タイプ **group** として ACL で指名されているグループのメンバーでない場合、および **user_obj** エントリーや **group_obj** エントリーが示すプリンシパルと一致しない場合に限ります。入力フォーマットは次のとおりです。

{**other_obj** *permissions*}

user ACL のデフォルト・セルの特定のプリンシパルの許可を確立します。この ACL エントリー・タイプは、プリンシパル名であるキーを必要とします。入力フォーマットは次のとおりです。

{**user** *principal_name permissions*}

group デフォルト・セルの特定のグループのメンバーのための許可を確立します。この ACL エントリー・タイプは、グループ名であるキーを必要とします。入力フォーマットは次のとおりです。

{**group** *group_name permissions*}

foreign_user
ACL のデフォルト・セル以外の外部セル内の特定のプリンシパルのための許可を確立します。キーとしてプリンシパル名とセル名を指定して、プリンシパルを識別する必要があります。入力フォーマットは次のとおりです。

{**foreign_user** *cell_name*/*principal_name* ¥ *permissions*}

foreign_group
ACL のデフォルト・セル以外の外部セルの特定のグループについての許可を確立します。キーとしてグループ名とセル名を指定して、グループを識別する必要があります。入力フォーマットは次のとおりです。

{**foreign_group** *cell_name*/*group_name permissions*}

foreign_other
特定の外部セル (ACL のデフォルト・セル以外のもの) 内の他のプリンシパルについての許可を確立します。これらのプリンシパルは、エントリー・タイプ **foreign_user** の ACL エントリー内で明示的に指名されていないもので、しかもタイプ **group_group** の ACL エントリー内で指名されている

グループのメンバーでないものです。キーとしてセル名を指定することによって、外部セルを識別する必要があります。入力フォーマットは次のとおりです。

{**foreign_other** *cell_name permissions*}

any_other

ローカル・セルまたは外部セル内の他のすべてのプリンシパルについての許可を確立します (ただし、それらが ACL 内のより特定化されたエントリーに一致しない場合に限る)。入力フォーマットは次のとおりです。

{**any_other** *permissions*}

user_obj_delegate

オブジェクトの実際のユーザーあるいは実効ユーザーに対する仲介者についての許可を確立します。入力フォーマットは次のとおりです。

{**user_obj_delegate** *permissions*}

group_obj_delegate

オブジェクトの実際のグループあるいは実効グループのメンバーに対する仲介者についての許可を確立します。入力フォーマットは次のとおりです。

{**group_obj_delegate** *permissions*}

other_obj_delegate

デフォルト・セルの他のすべてのプリンシパルに対する仲介者についての許可を確立します。ただし、それらがエントリー・タイプ **user** の ACL で特に指名されておらず、エントリー・タイプ **group** の ACL で指名されているグループのメンバーではなく、しかも **user_obj** エントリーや **group_obj** エントリーが示すプリンシパルに一致しない場合に限ります。入力フォーマットは次のとおりです。

{**other_obj_delegate** *permissions*}

user_delegate

ACL のデフォルト・セル内の特定のプリンシパルに対する仲介者についての許可を確立します。この ACL エントリー・タイプは、プリンシパル名であるキーを必要とします。入力フォーマットは次のとおりです。

{**user_delegate** *principal_name permissions*}

group_delegate

デフォルト・セル内の特定のグループのメンバーに対する仲介者についての許可を確立します。この ACL エントリー・タイプは、グループ名であるキーを必要とします。入力フォーマットは次のとおりです。

{**group_delegate** *group_name permissions*}

foreign_user_delegate

ACL のデフォルト・セル以外の外部セル内の特定のプリンシパルに対する仲介者についての許可を確立します。キーとしてプリンシパル名とセル名を指定して、プリンシパルを識別する必要があります。入力フォーマットは次のとおりです。

{**foreign_user_delegate** *cell_name*/*principal_name* ¥ *permissions*}

foreign_group_delegate

ACL のデフォルト・セル以外の外部セル内の特定のグループに対する仲介

者についての許可を確立します。キーとしてグループ名とセル名を指定して、グループを識別する必要があります。入力フォーマットは次のとおりです。

{**foreign_group_delegate** *cell_name/group_name ¥ permissions*}

foreign_other_delegate

特定の外部セル (ACL のデフォルト・セル以外のもの) 内の他のプリンシパルに対する仲介者についての許可を確立します。これらのプリンシパルは、エントリー・タイプ **foreign_user** の ACL エントリーで明示的に指名されていないもので、しかもタイプ **group_group** の ACL エントリーで指名されているグループのメンバーでないものです。キーとしてセル名を指定することによって、外部セルを識別する必要があります。入力フォーマットは次のとおりです。

{**foreign_other_delegate** *cell_name permissions*}

any_other_delegate

ローカル・セルあるいは外部セル内の他のすべてのプリンシパルが、 ACL 内のより特定化されたエントリーと一致しない場合に、それらのプリンシパルに対する仲介者についての許可を確立します。入力フォーマットは次のとおりです。

{**any_other_delegate** *permissions*}

グループ許可およびプロジェクト・リスト

プリンシパルは、プロジェクト・リスト (プリンシパルまたは別名がメンバーとなっているすべてのグループのリスト) からグループ許可を付加します。プリンシパルがオブジェクトへアクセスしようとするときに持っているアクセス権は、プリンシパルがメンバーとなっている、ACL にエントリーのある各グループに与えられた認可の、論理 OR から付加されたものです。プリンシパルがログインした名前と別名の両方ではなく、いずれか一方からのみプリンシパルが権利を付加することに注意してください。 (別名およびプロジェクト・リストの詳細については、 357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。)

たとえば、次に示すエントリーが ACL に含まれていると仮定します。

```
{user_obj crwxid-}
{group_obj crwx---}
{other_obj -r-----}
{group composers crwx---}
{user bach crwx---}
{user mozart crwx---}
{group performers --w-idt}
```

ユーザー **cole** は、グループ **composers** とグループ **performers** のメンバーです。 **cole** は両グループから許可を付加するので、そのアクセス許可は **crwxidt** となります。 (セキュリティー・サービスには、グループがプロジェクト・リストに含められないようにする方式があり、グループの許可がプロジェクト・リストの一部として付加されないようにします。詳細については、357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。)

プリンシパル ACL エントリーとグループ ACL エントリーの使用

セキュリティー・メカニズムによって ACL が適用される場合、 ACL エントリーは特定の順序で選択されます。最も特定化されたものが、特定化されていないものより先に選択されます。

プリンシパルとグループの ACL エントリー・タイプを使用する際に、 **user_obj**、**group_obj**、および **other_obj** タイプは、 POSIX ファイル許可の **user**、 **group** および **other** に類似していると考えてください。 **user** と **group** タイプは、特定のプリンシパルまたはグループの許可を指定するために使用します。

user_obj、**group_obj**、 **other_obj**、**user**、および **group** エントリー・タイプは、 ACL のデフォルト・セルのプリンシパルとグループに適用されます。外部セル内の特定のプリンシパルとグループに許可をセットするには、 **foreign_user** および **foreign_group** エントリーを使用します。これらのエントリーは、**user** および **group** エントリーがデフォルト・セルで行うのと同様の方法で、外部セルに許可をセットします。外部セル内のその他のものに許可をセットするには、**other_obj** がデフォルト・セル内の他のものに対して行う場合と同様に、**foreign_other** を使用します。

any_other エントリー・タイプは、他のエントリー・タイプが適用されない、すべてのローカルおよび外部のプリンシパルについて許可をセットします。他のいずれかのエントリー・タイプがローカルあるいは外部のプリンシパルに対して明示的あるいは暗示的にセットされている場合、 **any_other** エントリーは適用されません。これは、マネージャーがプリンシパルとエントリーの間に一致を検出すると、マネージャーは ACL リストの検査を停止して、検出したエントリー (グループの場合、複数のエントリー) を適用するためです。 ACL マネージャーは、**any_other** エントリー・タイプを調べる前に、他のすべてのエントリー・タイプ (以下で説明するマスク・タイプを除く) の中で一致するものがあるかどうかを調べます。 ACL 検査の順序の詳細については、 340ページの『ACL エントリーの検査順序』を参照してください。

マスクのための ACL エントリー・タイプ

ACL エントリー内のマスクは、プリンシパルに与えることができる最大の認可を確立します。マスクには、**mask_obj mask** と **unauthenticated mask** の 2 つがあります。 ACL エントリーとマスクで与えられた認可だけが認められます。たとえば、ACL エントリーが **rwx** 認可を指定し、マスクが **x** 認可だけを指定している場合、それらの認可とマスクは AND 条件にかけられて、 **x** 認可だけが与えられます。

mask_obj マスク (それが存在する場合) は、**user_obj** と **other_obj** を除く、すべてのエントリー・タイプに適用されます。 **unauthenticated** マスクは、認証されていないすべてのプリンシパルに適用されます。 ACL マネージャーは、ACL エントリーから許可を取り出すときに、 **mask_obj** マスク (存在する場合) でそれぞれをフィルターにかけ、最後に **unauthenticated** マスクのフィルターにかけて選別します。マネージャーは、最初に一致したエントリー、**mask_obj** マスク、および、**unauthenticated** マスクの認可だけを与えます。

注: **unauthenticated** マスクが作成されていない場合、認証されないプリンシパルは、オブジェクトへのすべてのアクセスを拒否されます。ユーザーが DCE 証

明書を持っていないために認証されない場合、そのユーザーと一致するエントリーは、**any_other** エントリー・タイプだけであり、これは **unauthenticated** マスクによってマスクされます。つまり、認証されないユーザーがオブジェクトにアクセスするには、オブジェクトの ACL に **any_other** エントリー・タイプと **unauthenticated** マスク・エントリーが含まれていなければならないことを意味します。

マスクの使用例を次に示します。ここでは、特定のオブジェクトについて、そのオブジェクトへの **rw** アクセスを指定する ACL エントリーが多数存在します。アクセスを一時的に読み取り専用に制限する必要がありますが、すべての ACL エントリーを変更したいわけではありません。この場合は、単に **r** の **mask_obj** マスクを作成し、終了後にそれを取り除くことによって、一時的な制限を行えます。

異なる DCE リリースのための ACL エントリー・タイプ

extended エントリー・タイプは、将来の DCE リリースが新しいエントリー・タイプをインストールできるようにするための、 ACL エントリーの汎用フォーマットを提供します。新しいタイプは **extended** エントリーの汎用フォーマットでパッケージされるため、以前の DCE リリースが新しいエントリー・タイプの意味を解釈できなくても、それらをコピー、表示、および印刷することはできます。

344ページの『ACL のコピー』は、拡張エントリーをコピーする方法を示しています。拡張エントリーは変更できないので注意してください。ただし、削除することはできます。

extended ACL エントリーの形式は次のとおりです。

{**extended** *uuid.ndr.ndr.ndr.ndr.number_of_bytes.data permissions*}

ここで、

uuid 拡張 ACL エントリーのエントリー・タイプを識別する UUID。 (この UUID は、本書に説明されている ACL エントリー・タイプの 1 つ、または現在未定義の ACL エントリー・タイプを識別できます。)

ndr.ndr.ndr.ndr
データのエンコードを識別する、ネットワーク・データ表現 (NDR) フォーマット・ラベル (16 進数形式、ドットで区切る)。

number_of_bytes
data のバイトの合計数を指定する 10 進数。この後に 1 つのドットが続きます。

data 16 進形式の ACL データ。 (ACL データの各バイトは、2 つの 16 進数字です。) ACL データには、(許可を除く) すべての ACL エントリー指定が含まれています。データが渡される ACL マネージャーがそのデータを理解できることが前提になっているため、 ACL データは解釈されません。

permissions
エントリーによって与えられる認可。

ACL エントリーの検査順序

ACL マネージャーは、ACL エントリーのリストを読み、特定の操作を実行しようとしている個人に適用される特定のエントリーを探します。まず ACL マネージャーは、アクセスを希望しているプロセスまたはプリンシパルの特権属性と一致する特権属性が ACL にリストされているかどうか探します。一致を見つけると、ACL マネージャーは一致する ACL エントリー内の許可を調べ、次に **mask_obj** マスクが存在すれば (それがエントリー・タイプ **user_obj** や **other_obj** でない限り)、**mask_obj** マスクをそれに適用します。最後に、ACL マネージャーは、プリンシパルが認証されていない場合は、 **unauthenticated** マスク (存在すれば) を適用します。要求されたアクセスを与える認可であれば、マネージャーは、プリンシパルにアクセスを与えます。そうでなければ、アクセスは拒否されます。

ACL マネージャーは一致を検出すると ACL エントリーの検査を停止するため、ACL 検査の順序を理解しておくことは重要です。 342ページの図50 は、検査の順序および適用されるマスクを示しています。 ACL マネージャーは、次の順序でエントリーを検査します。ただし、イニシエーターのプリンシパルは、 **..._delegate** エントリーに対して検査されません。代行プリンシパルは、すべてのエントリーに対して検査されます。

1. まず、ACL マネージャーは、以下の順序でユーザー ACL エントリーを検査します。
   - **user_obj**
   - **user_obj_delegate**
   - **user**
   - **user_delegate**
   - **foreign_user**
   - **foreign_user_delegate**

   ACL マネージャーは、最初にユーザー・エントリーの一致が見つかった時点ですべてのエントリー検査を停止し、エントリー内の許可を適用します。ユーザー・エントリーは、上記のリストに示されている順番で、最も特定化されたものから最も特定化されていないものへ検査されます。
2. ACL マネージャーは、ユーザー・エントリー内に一致するものが見つからないと、以下のグループ・エントリーのすべてを検査します。
   - **group_obj**
   - **group_obj_delegate**
   - **group**
   - **group_delegate**
   - **foreign_group**
   - **foreign_group_delegate**

   いずれかの ACL エントリーがプリンシパルのプロジェクト・リストと一致して、さらにこれらのエントリーからの認可を論理 OR した結果がアクセスを与える場合には、アクセスが許され、検査はそれ以上行われません。

   プリンシパルは、ACL にリストされたグループのうち、メンバーとなっている (プロジェクト・リスト内にある) すべてのグループから許可を付加します。このため、すべての グループが検査され、すべての プリンシパルのグループ許可について論理 OR が行われます。グループ・エントリーの検査の順番は、重要で

はありません。プロジェクト・リストの詳細については、 337ページの『グループ許可およびプロジェクト・リスト』を参照してください。

3. ACL マネージャーは、許可を求めているプリンシパルとグループ・エントリーのグループのメンバーとの間に一致を検出しないと、 **other_obj** および **other_obj_delegate** エントリーを検査します。一致を検出すると、ACL マネージャーは ACL エントリーの検査を停止します。
4. ACL マネージャーは、許可を求めているプリンシパルと **other_obj** または **other_obj_delegate** エントリーとの間に一致が見つからないと、**foreign_other** および **foreign_other_delegate** エントリーを検査します。一致を検出すると、ACL マネージャーは ACL エントリーの検査を停止します。
5. ACL マネージャーは、許可を求めているプリンシパルと **foreign_other** または **foreign_other_delegate** エントリーとの間に一致が見つからないと、**any_other** および **any_other_delegate** エントリーを検査します。 **any_other** または **any_other_delegate** エントリーに一致するものがないと、オブジェクトへのすべてのアクセスが拒否されます。

最終的な許可は、イニシエーター・プリンシパルと各代行の許可の交点となります。

342ページの図50 は、ACL エントリーに適用されるときの、これらのステップを示しています。 2 つのカラムは、**mask_obj** によってマスクされない ACL エントリーと、マスクされる ACL エントリーとを区別しています。

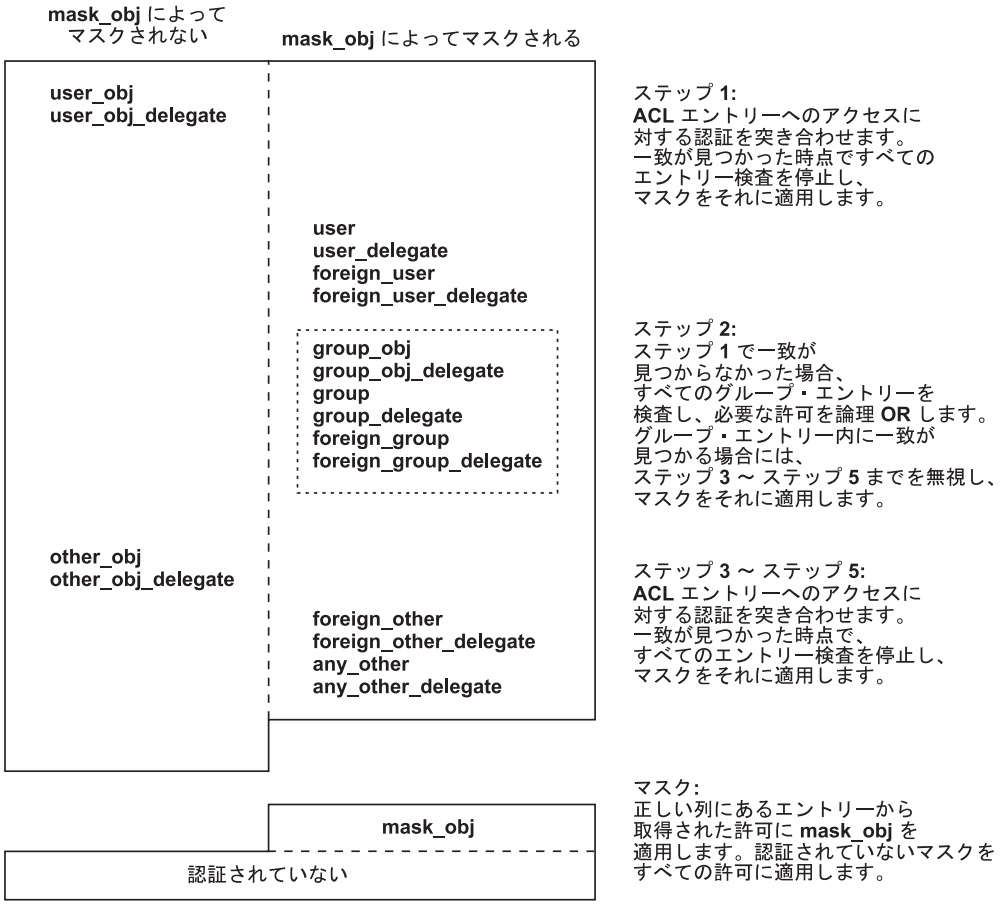


図 50. ACL 検査およびマスク適用の順序

mask_obj マスクと ACL 検査

ACL マネージャーは、ACL エントリーの検査から得られた認可を与える前に、**mask_obj** マスクを使用して、エントリーの認可をフィルターにかけます。 ACL エントリーおよびマスクで指名されている認可だけが与えられます。たとえば、ACL エントリーが **rwx** 認可を与え、 **mask_obj** エントリーが **r** と **w** の認可だけを指定している場合には、 **r** と **w** のみが認められます。 ACL エントリーで指定されている **x** 許可は、無視されます。

非認証マスクと ACL 検査

ACL マネージャーは、非認証プリンシパルからアクセス要求を受け取ると、 ACL エントリーを調べ、**mask_obj** マスクが有効であれば、上記のようにそれを適用します。次に、その結果得られた許可を、非認証プリンシパル (エントリー・タイプ **unauthenticated**) のマスクを使用してフィルターにかけます。 **unauthenticated** マスク、ACL エントリー、および **mask_obj** マスク (存在する場合) で指定されている認可だけが与えられます。

認可の発行に対する検査順序の影響

ACL エントリーの検査は、最も特定化されたものから最も特定化されていないものへと行われます。たとえば、ACL に次のエントリーが含まれているものと仮定します。

```
{user mahler r}
{group composers rwx}
```

グループ **composers** のメンバーである、 **mahler** という名前のプリンシパルが実行アクセス (**x**) を要求すると、それは拒否されます。これは、検査順序により、すべてのユーザー・エントリー (**user_obj**、 **user**、および **foreign_user**) が検査されてから、すべてのグループ (**group_obj**、**group**、および **foreign_group**) のエントリーが検査されるためです。この結果、ACL マネージャーが見つける最初の一致は、ユーザー **mahler** とユーザー **mahler** の ACL エントリーの間の一致です。ユーザー・エントリーの一致が検出されると、検査は停止されて、検出された許可が適用されます。この場合、より自由な許可を持つエントリーである、 **group** エントリーの前で検査が停止します。

アクセスの拒絶

プリンシパルあるいはグループのための ACL エントリーを作成するとき、 ACL エントリーで指定した認可だけを与えることになります。あるプリンシパルによる、あるオブジェクトへのアクセスをすべて拒絶するには、許可の代わりにダッシュが入った ACL エントリーを作成します。たとえば、ユーザー **mozart** のすべてのアクセスを拒絶する場合、エントリーは次のようになります。

```
{user mozart -}
```

特定のプリンシパルあるいはグループに対するアクセスを拒絶する場合は、使用可能な最も特定化されたエントリー・タイプを選択します。プリンシパルの場合、これは一般的には **user** または **foreign_user** のエントリー・タイプであり、グループの場合、**group** または **foreign_group** のエントリー・タイプです。プリンシパルがオブジェクトの所有者、またはオブジェクトのグループのメンバーである場合に、アクセスが拒絶されるようにするには、エントリー・タイプ **user_obj** または **group_obj** を使用しなければなりません。

すべての非認証ユーザーのアクセスを拒絶する場合は、 **unauthenticated** マスクを作成しないでください。このマスクが作成されない場合 (**unauthenticated** の ACL エントリー・タイプ)、オブジェクトをアクセスできるのは認証されたプリンシパルだけです。これと同じ効果は、許可のない (つまり許可の代わりにダッシュを使った) **unauthenticated** マスクを作成することによっても得られます。この方法には、非認証ユーザーがアクセス権を持たないということを視覚的に示せるという利点もあります。

ACL 管理作業

ACL 管理には、DCE エンティティー上の ACL エントリーの作成、変更、および削除操作が含まれます。これらの作業はすべて DCE 制御プログラムを使用して行うことができます。制御プログラムの **acl** コマンドは、 ACL に対して以下の操作を実行します。

- ローカル・セルおよび外部セル内の DCE オブジェクトの ACL エントリーの作成と変更。 (オブジェクトは、作成時に初期 ACL エントリーに関連付けられることに注意してください。詳細については、『ファイルからの ACL の生成』を参照してください。)
- オブジェクトの ACL マネージャーがオブジェクトに実装した許可の表示。
- 可能な許可を制限するために使用する、マスクの作成と変更。

注: UNIX モードの表示および操作を行うための標準の UNIX ツールは、使用しているファイル・システム用に設定されている ACL にのみ影響します。

DCE 制御プログラムの **acl** コマンドの詳細説明については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

ACL のコピー

1 つの DCE オブジェクトから別の DCE オブジェクトへ ACL をコピーするには、次に示すように、**-acl** オプションを指定して、 DCE 制御プログラム **acl replace** コマンドを実行します。

```
dcecp> acl replace  /.:/hosts/hermes  -acl [acl show /.:/hosts/cyclops]
dcecp>
```

このコマンド例では、ホスト **hermes** の ACL をホスト **cyclops** の ACL に置き換えます。後者は、 **-acl** オプションによって呼び出される **acl show** コマンドに名前が指定されています。 **-acl** オプション内の **-acl show** コマンドを [ ] (大括弧) で囲んでいることに注意してください。これは、**-acl** オプション値がコマンド呼び出しの場合に必要です。

セル間でコピーを行う場合は、 **acl replace** コマンドの **-cell** オプションを、 **-acl** オプションと同様に使用します。たとえば、次のようにします。

```
dcecp> acl replace /.:/hosts/hermes -acl [acl show /.:/hosts/cyclops]
¥
> -cell [acl show /.:/hosts/cyclops -cell]
dcecp>
```

1 つの ACL マネージャーのドメインから別の ACL マネージャーのドメインへ **extended** エントリー・タイプをコピーするには、 **acl replace** コマンドへの入力として、 **dcecp acl show** コマンドの出力を使用します。この方法で **extended** エントリーをコピーするには、両方の ACL マネージャーが **extended** エントリー・タイプをサポートしている必要があります。

ファイルからの ACL の生成

ACL を手早く作成するには、必要な ACL エントリーを含むテキスト・ファイルを作成して編集し、それから **acl replace** コマンドを使用して ACL を生成します。

たとえば、ファイル **std_acl** に次のエントリーが含まれているとします。

```
mask_obj:crwxid-
user_obj:crwxid-
group_obj:crwx---
other_obj:-r-----
```

```
user:lizt:crwx---
group:composers:-r-----
user:bach:crwx---
user:mozart:crwx---
```

次の **acl replace** コマンドは、 **std_acl** 内のエントリーを **/.../dresden.com/my_filesystem/opus** という名前の ACL に追加します。

```
dcecp> acl replace /.../dresden.com/my_filesystem/opus -acl [cat
std_acl]
dcecp>
```

acl replace コマンドは、ファイル **std_acl** からのエントリーで、すべての ACL エントリーを上書きします。以前の内容にかかわらず、 **opus** の ACL は次のようになります。

```
mask_obj:crwxid-
user_obj:crwxid-
user:lizt:crwx---
user:bach:crwx---
user:mozart:crwx---
group_obj:crwx---
group:composers:-r-----
other_obj:-r-----
```

コンテナー ACL

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

オブジェクト ACL は、オブジェクト自体へのアクセスを制御します。コンテナー・オブジェクトは、オブジェクト ACL に加えて、初期コンテナー ACL と初期オブジェクト ACL を保持しています。これら 2 つの ACL は ACL としてアクセス制御に使用されず、初期コンテナー内で作成されたオブジェクトまたはコンテナーの初期 ACL のクローンに使用されます。初期コンテナー ACL と初期オブジェクト ACL は、 **dcecp acl** コマンドで **-ic** および **-io** オプションを使用することによって、通常の ACL と同様に編集することができます。

オブジェクトとコンテナー

オブジェクトに使用される ACL のタイプは、オブジェクトが単純オブジェクトであるか、コンテナーであるかによって決まります。コンテナーとは、他のオブジェクトを保持するオブジェクトです。保持するオブジェクトは、単純なオブジェクトまたはコンテナー・オブジェクトのいずれかです。単純オブジェクトは、他のオブジェクトを保持しません。いずれの DCE コンポーネントもオブジェクトとコンテナーを持つことができますが、最も単純で、また最も一般的な例は、ファイル・システムです。ファイル・システムには、ファイルとディレクトリーが存在します。ファイルが単純オブジェクトで、ディレクトリーがコンテナーです。ディレクトリーは、単純オブジェクト (ファイル) および他のコンテナー (サブディレクトリー) を保持することができます。

オブジェクト ACL は、単純オブジェクトおよびコンテナー・オブジェクトに関連付けられます。初期コンテナーと初期オブジェクト ACL は、コンテナー・オブジェクトにのみ関連付けられます。

オブジェクトとコンテナーの初期 ACL

初期 ACL エントリーおよびそれらを含む ACL は、オブジェクトが作成されるときに自動的に適用されます。エントリーは、DCE 制御プログラムを使用して、いつでも変更することができます。コンテナーとオブジェクトの初期 ACL として使用される DCE ACL のタイプは、次のとおりです。

- 初期コンテナー ACL は、コンテナー内に作成されるコンテナーのデフォルト ACL を決定します。たとえば、ディレクトリーのファイル・システム初期コンテナー ACL は、このディレクトリー内に作成されるサブディレクトリーのデフォルト ACL を指定します。
- 初期オブジェクト ACL は、コンテナーの中に作成されるオブジェクトのデフォルトを決定します。たとえば、ディレクトリーのファイル・システム初期オブジェクト ACL は、そのディレクトリー内に作成されるファイルのためのデフォルト ACL を指定します。

オブジェクトのデフォルト ACL

単純オブジェクトは、コンテナーの中に作成されるときに、コンテナーの初期オブジェクト ACL をオブジェクト ACL として継承します。 図51 は、コンテナーに作成される単純オブジェクトに、デフォルト ACL を割り当てる方法を示しています。

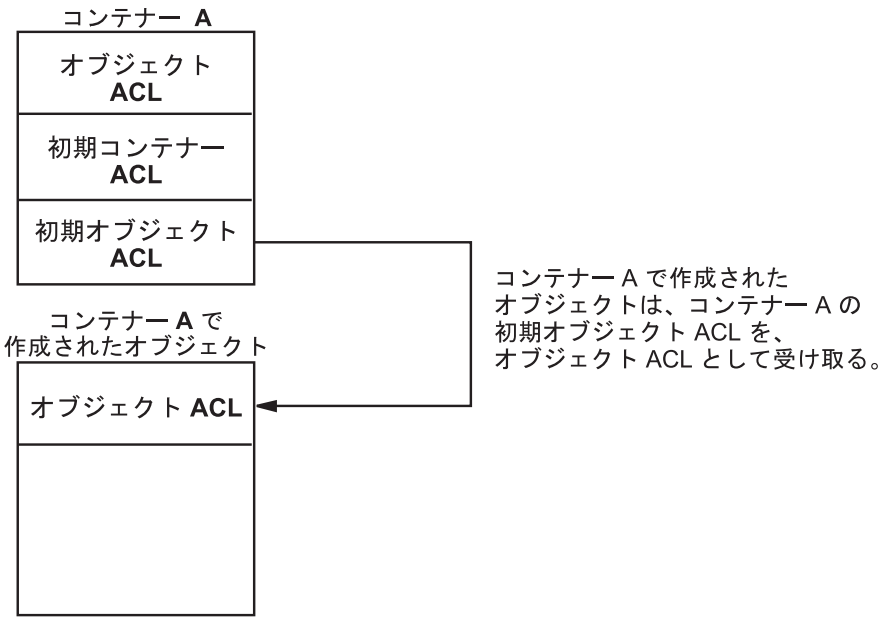


図 51. コンテナーに作成されるオブジェクトの初期 ACL

コンテナーのデフォルト ACL

コンテナー内にコンテナーが作成されると (たとえばディレクトリー内のサブディレクトリー)、そのコンテナーは親コンテナーから以下のものを継承します。

- 初期コンテナー ACL を、そのオブジェクト ACL およびその初期コンテナー ACL として継承。
- 初期オブジェクト ACL を、その初期オブジェクト ACL として継承。

たとえば、ディレクトリー **marketing** の中に **report** という名前のファイルを作成すると、システムは、ディレクトリー **marketing** の初期オブジェクト ACL を **report** に割り当てます。 **marketing** にサブディレクトリーを作成すると、システムは、**marketing** の初期コンテナー ACL を新しいサブディレクトリーに割り当てます。新しいサブディレクトリーは、親ディレクトリーの初期 ACL に一致する、一組の初期 ACL も受け取ります。この例では、新しいサブディレクトリーの ACL として、 **marketing** の初期 ACL も受け取ります。 図52 は、コンテナー内に作成されるオブジェクトにデフォルト ACL を割り当てる方法を示しています。

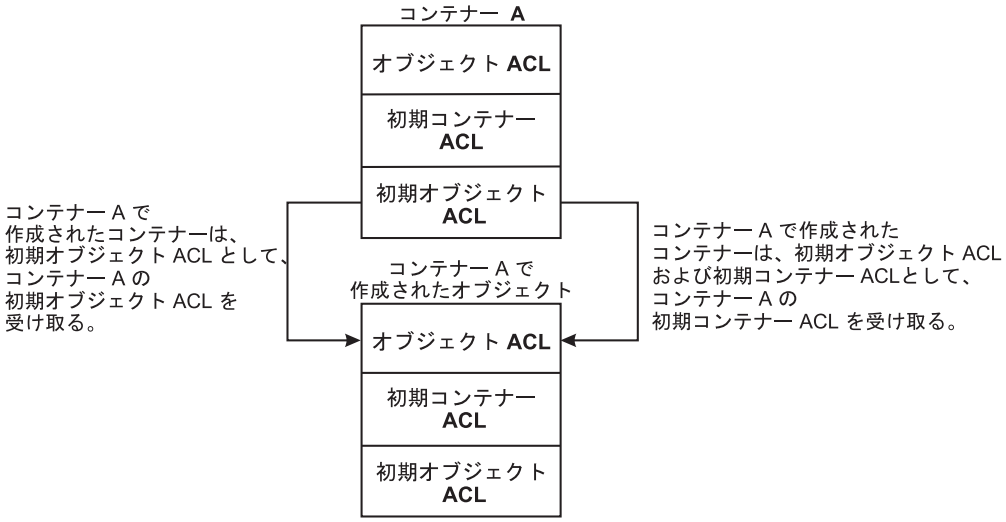


図 52. コンテナー内に作成されるコンテナーの初期 ACL

デフォルト・コンテナー ACL の例

以下の例では、コンテナーの中に作成されたコンテナーに最初に ACL を割り当てる方法を示します。

コンテナー A には次の ACL が含まれているものと仮定します。

オブジェクト ACL

```
{user_obj crwxid}
{group_obj crwxid}
{other_obj r}
```

初期コンテナー ACL

```
{user_obj crwxid}
{group_obj rw}
{other_obj r}
```

初期オブジェクト ACL

```
{user_obj crwxid}
{group_obj r}
{other_obj r}
```

コンテナー B がコンテナー A の中に作成されるとき、コンテナー B は次に示すデフォルト ACL を保持しています。

オブジェクト ACL (コンテナー A の初期コンテナー ACL)

```
{user_obj crwxid}
{group_obj rw}
{other_obj r}
```

初期コンテナー ACL (コンテナー A の初期コンテナー ACL)

```
{user_obj crwxid}
{group_obj rw}
{other_obj r}
```

初期オブジェクト ACL (コンテナー A の初期オブジェクト ACL)

```
{user_obj crwxid}
{group_obj r}
{other_obj r}
```

ACL 編集時のマスクの影響

ユーザーが新しい **mask_obj** ACL エントリーを指定すると、 **acl modify** はそのエントリーを使用します。指定しない場合、**acl modify** コマンドは、ユーザーが **-mask** calc、**-mask** nocalc、または **-purge** オプションのいずれかを指定していなければ、次の段落に示すアルゴリズムを使用してマスクを計算し直します。したがって、マスクを変更することによって、各 **acl modify** コマンドでより多くの認可またはより少ない認可を与えることができます。

acl modify コマンドがマスクを計算するときに使用するアルゴリズムを以下に示します。

1. ファイルの既存の ACL を検索します。
2. エントリーを除去して既存の許可を減らすための、すべての要求を実行します。
3. 残りすべてのエントリーの実許可の合併を計算します。
4. 実権利と実効権利とでどの許可が異なるかを調べます。 (これは、ステップ 3 と 4 の結果を論理 XOR したものです。)
5. ACL に新しいエントリーを追加するすべての要求、および既存のエントリーの許可を増やすすべての要求を実行します。
6. これらの新たに加えられた認可と、ステップ 4 からの古い実効認可の合併を計算します。これが新しいマスク値の候補になります。
7. 候補の新しいマスクに認可が存在し、その認可がさらに、元の実際の権利と (ステップ 5 からの) 実効権利との間で異なる認可の中にもある場合、候補の新しいマスクを適用すると、ユーザーの意図しない新しい権利が予期せずに与えられることになります。ユーザーがオプション **-mask calc**、**-mask nocalc**、または **-purge** のいずれかを指定しない限り、この条件はエラーとなり、ACL は変更されません。オプションを指定していれば、候補の新しいマスクが、新しいマスクとして適用されます。

大部分の ACL 操作では、このような自動再計算が安全です。まれに、マスクの再計算によって、ユーザーが予期しない追加の権利が与えられることがあります。たとえばユーザーが指定せず、そのエントリーの以前の実効権利にも存在しなかった認可が、エントリーに与えられることがあります。

次の例は、マスクの再計算の方法とオプションの影響を示しています。

あるユーザーに **rwx** 認可を与えるエントリーが ACL に含まれていますが、マスクでは **r-x** の実効認可を認めているとします。新しい **rwx** ACL エントリーを追加してマスクを (ステップ 6 にしたがって) **rwx** に再計算するのは危険です。なぜなら、最初のユーザーの実効アクセス権が、予期せずに **r-x** から **rwx** に変更されるためです。 **acl modify** コマンドがこのような危険な状況を検出すると、そのデフォルトのアクションはエラー・メッセージを出し、ACL を変更しません。

許可および実効許可を示す初期状態は、次のようになります。

```
dcecp> acl show /.:/concertos
{user vivaldi rwx effective r-x}
{mask_obj r-x}
dcecp>
```

次に示すようにユーザーを追加すると、マスクの再計算により、 **vivaldi** には **rwx** の実効許可が与えられることになるため、エラーになります。

```
dcecp> acl modify /.:/concertos -add {user telemann rwx}
Error: Unintended permissions not granted.
dcecp>
```

-mask calc オプションを明示的に使用すると、 **vivaldi** に与えられた新しい認可を無視して、再計算されたマスクが適用できます。マスクは、ACL のファイル・グループ・クラス・エントリーに与えられた認可の合併にセットされます。このオプションを使用すると、余分な認可を不注意に与える可能性があります。

```
dcecp> acl modify /.:/concertos -add {user telemann
rwx} -mask calc
dcecp> acl show /.:/concertos
{user vivaldi rwx effective rwx}
{user telemann rwx effective rwx}
{mask_obj rwx}
dcecp>
```

-mask nocalc オプションを明示的に使用すると、 **r-x** マスクが保持され、**telemann** の実効許可を減らす結果となります。 ACL はユーザーが指定した通りに変更され、マスクの再計算および許可の除去は行われません。

```
dcecp> acl modify /.:/concertos -add {user telemann rwx} -mask nocalc
dcecp> acl show /.:/concertos
{user vivaldi rwx effective r-x}
{user telemann rwx effective r-x}
{mask_obj r-x}
dcecp>
```

-purge オプションを使用すると、すべてのエントリーの実許可が実効許可で置き換えられます。厳密に言うと、安全でない条件をコマンドが検出すると、この条件によって、ファイル・グループ・クラス内にある変更されていないすべてのエントリーとマスクの現在値が交差し、すべての ACL エントリー (ただし **user_obj**、**other_obj**、 **mask_obj** および **unauthenticated** を除く) がそれらの実効許可で置き換えられます。

```
dcecp> acl modify /.:/concertos -add {user telemann rwx} -purge
dcecp> acl show /.:/concertos
{user vivaldi rwx effective r-x}
{user telemann rwx effective rwx}
{mask_obj rwx}
dcecp>
```

第29章 DCE セキュリティー・サービスを管理する制御プログラム

DCE セキュリティー・サービスのための管理作業のほとんどは、 DCE 制御プログラム (**dcecp**) を使用して行うことができます。ただし、このサービスのコンポーネントのいくつかについては、 DCE に用意されている、別の制御プログラムを使用する必要があります。

ここでは、DCE セキュリティー・サービスの管理用に DCE 制御プログラムが提供するコマンドについての情報を提供します。また、この章では、レジストリー・エディター・プログラム (**rgy_edit**) がローカル・レジストリーの保守用に提供するコマンドについても説明します。

password_export や **sec_create_db** などの、セキュリティー関連の管理作業に時折使用する制御プログラムについては、ここでは説明していません。それらのプログラムについては、作業の実行方法とともに、本書の後の部分で説明しています。

DCE 制御プログラムの使用

DCE 制御プログラムとそのコマンド構文についての詳細情報は、本書の1ページの『第1部 DCE 制御プログラム』にあるので、ここではそれらの情報については繰り返しません。ここでは、 DCE 制御プログラムが DCE セキュリティー・サービスの管理のために提供する特定のコマンドについてのみ説明します。

DCE 制御プログラムは、 DCE セキュリティー・サービスのネットワーク全体のレジストリーのために、プリンシパル、グループ、オーガニゼーション、およびアカウントを作成し、保守します (レジストリー・サービス・コンポーネント)。制御プログラムは、ローカル・ノード上のセキュリティー・サーバーのパスワードを保護する、 keytab ファイルも操作します (認証サービス・コンポーネント)。そのほかに、DCE リソースを保護する ACL を保守します (特権サービス・コンポーネント)。 DCE セキュリティー・サービスを管理するための DCE 制御プログラム・コマンドは、コマンドが定義する各種のオブジェクトを通して、これらのセキュリティーと DCE 全体のリソースを操作します。たとえば、制御プログラムの **acl check** コマンドは、 DCE セキュリティー・サービス・オブジェクトの ACL が呼び出し側のプリンシパルに与える認可を表示します。

次のサブセクションでは、 DCE 制御プログラムが操作する DCE セキュリティー・サービス・オブジェクトと、制御プログラムがこれらのオブジェクトに対して行う操作のタイプについて説明します。

セキュリティー・サービス・オブジェクト

DCE 制御プログラムは、次に示すセキュリティー・サービス・コンポーネントを操作する機能を備えています。

プリンシパル
このオブジェクトは、レジストリーのプリンシパルを表します。これらのプ

リンシパルとしては、ネットワークの人間のユーザー、ネットワーク上のサーバー、ネットワーク上のマシン、あるいはローカル・セルがセル間の認証を行うセルがあります。

グループ
このオブジェクトは、レジストリーのグループを表します。グループは、オブジェクトへのアクセス権を割り当てる対象になりうるプリンシパルの集まりです。

オーガニゼーション
このオブジェクトは、レジストリーのオーガニゼーションを表します。オーガニゼーションとは、管理領域を拡大するためにポリシーを割り当てる対象となりうる、プリンシパルの集まりです。

アカウント
このオブジェクトは、プリンシパルのレジストリーに確立されるアカウントを表します。

レジストリー
このオブジェクトは、DCE セル内のレジストリー、つまり、DCE セキュリティー・サービスのアカウント情報のデータベースを表します。操作されるレジストリー・コピーは、マスター・レプリカまたはスレーブ・レプリカのいずれかです。

xattrschema
このオブジェクトは、DCE セキュリティー・サービス・コンポーネント、およびローカル・ホスト上のホスト・デーモン (**dced**) が保持するデータに対して指定する、拡張レジストリー属性 (ERA) のスキーマ (つまり定義) を操作します。

acl このオブジェクトは、 DCE セキュリティー・サービスの ACL 機能によって保護される、すべての DCE エンティティーの ACL を表します。

keytab
このオブジェクトは、DCE セキュリティー・サービスの認証されたサーバー・プリンシパルのキー (またはパスワード) を保管する、ファイルを表します。

DCE セキュリティー・サービスのための DCE 制御プログラムの操作

表16 は、 **dcecp** が DCE セキュリティー・サービス・オブジェクトに対して実行する操作のリストです。

プリンシパル、グループ、オーガニゼーション、およびアカウントを作成し保守するための DCE 制御プログラム・コマンドの使用については、本書の357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』と 395ページの『第31章 アカウントの作成および保守』で説明しています。

表 16. DCE セキュリティー・サービスに対する DCE 制御プログラムの操作

操作	説明
add	プリンシパル、グループ、またはオーガニゼーションをレジストリー・レプリカに追加します。

表 16. DCE セキュリティー・サービスに対する DCE 制御プログラムの操作 (続き)

操作	説明
catalog	レジストリー・レプリカ内のプリンシパル、グループ、およびオーガニゼーションのすべての名前を表示します。レジストリー自体について、 DCE セルに存在するマスター・レプリカとスレーブ・レプリカを表示します。
check	DCE ACL が現在のところセキュリティー・プリンシパルに与えている認可を表示します。
checkpoint	レジストリー・チェックポイント・インターバルをリセットします。
create	レジストリー・レプリカに新しいプリンシパル、グループ、オーガニゼーション、またはアカウントを作成します。また、ERA スキーマの新しいエントリーも作成します。
delete	レジストリー・レプリカからプリンシパル、グループ、オーガニゼーション、またはアカウントを削除します。レジストリー自体については、スレーブ・レプリカを削除します。 ERA スキーマの場合、エントリーを削除します。 DCE ACL の場合、ACL エントリーを除去します。
designate	どのレジストリー・レプリカがマスターであるかを変更します。
destroy	レジストリー・データベースの指定されたレプリカとそのコピーを削除します。
disable	レジストリーのマスター・レプリカの更新を不能にします。
dump	セルに存在するレジストリーの各レプリカの情報を表示します。
enable	レジストリーのマスター・レプリカの更新を可能にします。
generate	既存のレジストリー・アカウントのためのランダム・パスワードを生成します。
help	レジストリー・レプリカのプリンシパル、グループ、オーガニゼーション、アカウント、ERA スキーマまたは DCE ACL について、あるいはレジストリー・レプリカ自体についてのヘルプ情報を表示します。
list	レジストリー・レプリカ内のグループまたはオーガニゼーションに属するプリンシパルの名前を表示します。
modify	プリンシパル、グループ、アカウント、ERA スキーマ・エントリー、 DCE ACL エントリー、またはレジストリー自体のレジストリー・レプリカの中の属性情報を変更します。オーガニゼーションについては、ポリシー情報も変更します。
operations	プリンシパル、グループ、オーガニゼーション、アカウント、ERA スキーマ、 DCE ACL、またはレジストリー・レプリカが行う、あるいはそれらに対して行うことができる、操作を表示します。
permissions	保護された DCE コンポーネント上の ACL によって与えられた、認可を表示します。
remove	レジストリー・レプリカ内のグループまたはオーガニゼーションから、 1 つまたは複数のプリンシパルを除去します。
rename	レジストリー・レプリカ内のプリンシパル、グループ、オーガニゼーション、または ERA スキーマの名前を変更します。

表 16. DCE セキュリティー・サービスに対する DCE 制御プログラムの操作 (続き)

操作	説明
replace	DCE コンポーネント上の ACL 全体、またはレジストリー・レプリカのアドレスを置換します。
show	プリンシパル、グループ、ERA スキーマ・エントリー、または DCE ACL エントリーの属性についての情報を表示します。また、オーガニゼーション、アカウント、またはレジストリー・レプリカのポリシーについての情報も表示します。
stop	セキュリティー・サーバー・プロセスを停止します。
synchronize	レジストリーのスレーブ・レプリカに対して、その内容をマスター・レプリカから更新するよう指示します。
verify	すべてのレジストリーのレプリカが最新であるかどうか調べます。

レジストリー・エディターの使用

DCE 制御プログラムを使用してネットワーク全体のレジストリーを保守することはできますが、 1 つのセル内のホストにあるローカル・レジストリーを保守するには、レジストリー・エディターを使用する必要があります。

以下のサブセクションでは、レジストリー・エディターの開始、停止、およびヘルプの表示方法について説明し、ローカル・レジストリーの保守のために使用するコマンドについて述べます。レジストリー・エディターを使用してローカル・レジストリーを保守する具体的な方法については、 357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』および 395ページの『第31章 アカウントの作成および保守』を参照してください。

すべてのレジストリー・エディター・コマンドの詳細については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

開始、停止、およびヘルプの表示

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

レジストリー・エディターは、対話モードとコマンド行モードの 2 つのモードで実行できます。対話モードでは、制御プログラムが必要な情報の入力をユーザーに求めます。コマンド行モードでは、制御プログラムが必要とするすべての情報を、ユーザーがコマンド行で入力します。コマンド行モードでは一度に 1 つの操作しか実行できません。しかし、コマンド行モードは、一連のレジストリー・エディター・コマンドを実行するシェル・スクリプトの作成には便利です。本書のほとんどの例は対話モードの例です。 (レジストリー・エディターをコマンド行モードで起動して使用する方法については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。)

対話モードでレジストリー・エディターを起動するには、次のコマンドを入力します。

```
$ dceshared/bin/rgy_edit
```

注: ユーザーが認証されていない場合、**rgy_edit** コマンドは、バインディングが認証されていないことを警告する旨の次のようなメッセージを出します - `Can't establish authentication to registry`。

これに応答して、レジストリー・エディターは、現在のレジストリー・サイトの名前と **rgy_edit=>** プロンプトを次のように表示します。

```
Current site is:
registry server at /.../bayre.com/subsys/dce/sec/oddball
rgy_edit=>
```

ネーム・サービスが名前を提供できない場合は、出力は短縮されます。たとえば、以下のように出力するかわりに、

```
registry server at /.../bayre.com/subsys/dce/sec/oddball
```

表示は次のようになります。

```
registry server at /.../bayre.com
```

rgy_edit コマンドを終了するには、コマンド・プロンプトで **Return** を押します。たとえば、プリンシパルを追加する **add** コマンドを終了するには、 **Add Principal=> Enter name:** プロンプトで **Return** を押します。

レジストリー・エディターを終了するには、 **rgy_edit** プロンプトで **q[uit]** コマンドを入力します。

```
rgy_edit=> q
$
```

rgy_edit help コマンドはヘルプ情報を表示します。 **help** または **h** を入力すると、レジストリー・エディターはすべてのコマンドと表示可能なトピックのリストを表示します。たとえば、次のようにします。

```
rgy_edit=> help
```

ローカル・レジストリー保守のための rgy_edit コマンド

ローカル・レジストリーを表示したり、ローカル・レジストリーに対して保守作業を実行するには、まずそれにアクセスしなければなりません。ローカル・レジストリーにアクセスするには、次のように、 **-l** オプションを指定してレジストリー・エディターを呼び出します。

```
$ rgy_edit -l
rgy_edit=>
```

rgy_edit プロンプトで、実行したい特定の操作のコマンド名を入力します。 表17は、使用可能なコマンドをリストしています。

表 17. ローカル・レジストリーの保守のための rgy_edit コマンド

コマンド	機能
del[**ete**]	プリンシパル、グループ、またはオーガニゼーションのエントリーを削除します。

表 17. ローカル・レジストリーの保守のための *rgy_edit* コマンド *(続き)*

コマンド	機能
prop[**erties**]	ローカル・レジストリーの属性を表示または変更します。
pu[**rge**]	プリンシパル、グループ、またはオーガニゼーションの有効期限が切れたエントリーを削除します。
v[**iew**]	プリンシパル、グループ、またはオーガニゼーションのエントリーを表示します。

すべてのレジストリー・エディター・コマンドの詳細については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

第30章 プリンシパル、グループ、および組織の作成および保守

ここでは、**dcecp** を使用して、プリンシパル、グループ、オーガニゼーションを作成および保守する方法を説明します。始めにプリンシパル、グループ、オーガニゼーションに割り当てられる名前について説明し、続いて、レジストリー・オブジェクトを識別するために DCE セキュリティー・サービスが内部的に使用する汎用固有 ID (UUID) について説明します。

プリンシパル、グループ、およびオーガニゼーションの名前

レジストリー内の各プリンシパル、グループ、およびオーガニゼーションに対して名前を割り当てる必要があります。プリンシパル、グループ、およびオーガニゼーションに同じ名前を付けることはできますが、プリンシパル同士、 グループ同士、あるいはオーガニゼーション同士を同じ名前にすることはできません。たとえば、2 つのプリンシパルに **smith** という名前を付けることはできませんが、プリンシパルに **smith** と名前を付け、グループに **smith** と名前を付け、オーガニゼーションに **smith** と名前を付けることはできます。

1 次名、フルネーム、および別名の、3 タイプまでの名前を割り当てることができます。

1 次名

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

1 次名は、プリンシパル、グループ、およびオーガニゼーションに割り当てられます。レジストリー・オブジェクトの 1 次名は、人が読むことができるストリングが必要な場合に、ほとんどのシステム・ユーティリティーで使用される名前です。レジストリー・データベースにプリンシパル、グループ、またはオーガニゼーションを追加する場合、1 次名を与える必要があります。 1 次名は、レジストリー・データベースへの照会に使う **principal show** コマンドへの入力として使用できるキー・フィールドです。

フルネーム

オプションとして、フルネームをプリンシパル、グループ、およびオーガニゼーションに割り当てることができます。オブジェクトのフルネームは情報を与えるためのものです。一般に、フルネームは 1 次名を拡張したり説明したりして、ユーザーが認識しやすいようにします。たとえば、プリンシパルの 1 次名を **jsbach**、フルネームを **Johann S. Bach** とすることができます。また、オーガニゼーションの 1 次名を **moco**、フルネームを **Motet Composers** とすることができます。

フルネームは、データ・フィールドのみです。レジストリー・データベースを照会するためにこれを使用することはできません。プリンシパル、グループ、およびオーガニゼーション自体を作成する際に、プリンシパル、グループ、またはオーガニゼーションのフルネームを作成できます。

別名

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

別名は、1 次名のオプショナルの代替名です。別名は、プリンシパルとグループに割り当てることができますが、オーガニゼーションには割り当てることができません。別名と、その 1 次名とは、同じ UUID と UNIX ID を共用します。 (UUID と UNIX ID については、360ページの『汎用固有 ID と UNIX ID』で説明されています。) 別名は、レジストリー・データベースの照会のために使用できるキー・フィールドです。

それぞれの 1 次名に 1 つのアカウントを作成し、それぞれの別名にも 1 つのアカウントを作成することができるので、別名には同じプリンシパルに対して複数のアカウントを確立できるという柔軟性があります。たとえば、1 次名 **mahler** に、**gustav**、**gus**、および **gm** という 3 つの別名を作成するとします。こうすると、プリンシパル **mahler** について 4 つのアカウント、つまり 1 次名に 1 つと、名前の別名ごとに 1 つのアカウントの計 4 つを作成することができます。これらのアカウントには、別々のホーム・ディレクトリーとパスワードを使用することができ、また別々グループとオーガニゼーションに関連付けることができます。

プリンシパルは、ログインに使用する 1 次名または別名に関連する権利だけを獲得するので、同じユーザーに対してこれらの複数のアカウントを使用し、異なるアクセス・パターンをもたせることができます。たとえば、 **mahler** を **composers** グループのメンバーとし、また **gustav** を、システム管理者からなるグループである **music_admin** グループのメンバーとすることができます。プリンシパル **mahler** は、日常の作業を実行するには **mahler** としてログインし、管理上の仕事を実行するには **gustav** としてログインします。システムへの偶発的損傷を避けるためには、ユーザーがタスクを実行するときに必要最小限の特権でアカウントにログインできるようにアカウントをセットアップするのが賢明です。

グループについては、 2 つのグループ名を同じ UNIX 番号に関連付けたいときに別名が便利です。

別名の作成については、393ページの『プリンシパルまたはグループの別名の作成および保守』を参照してください。

名前のフォーマット

レジストリーの名前には、@ (アットマーク) と : (コロン) 以外の、あらゆる文字や数字を含めることができます。ただし、名前にはスペース、{} (中括弧)、および 『』 (引用符) を使用しないことを推奨します。これらを含めると、 **dcecp** が名前を正しく解析できないことがあるためです。

名前に使用できる最大文字数は、1024 です。

予約済みプリンシパルおよびアカウント

各種のシステム操作用にいくつかのプリンシパルとアカウントが予約されています。予約されているプリンシパルを削除することはできません。予約されているアカウントは、変更することはできますが、直接、削除することはできません。ただし、アカウントに指定されているグループまたはオーガニゼーションを削除することによって、予約されているアカウントを間接的に削除できます。 (詳細は、395ページの『第31章 アカウントの作成および保守』を参照してください。)

予約済みのプリンシパルとアカウントのリストを次に示します。リスト内の *cell_name* は、セルの名前です。

- 予約済みのプリンシパル:
  - **dce-ptgt**
  - **krbtgt/***cell_name*
  - **dce-rgy**
- 予約済みのアカウント:
  - **dce-ptgt none none**
  - **krbtgt/***cell_name* **none none**
  - **dce-rgy none none**

オブジェクト作成割り当て量

各プリンシパルにオブジェクト作成割り当て量を割り当てることができます。この割り当てにより、プリンシパルが作成できるレジストリー・オブジェクトの数を制御できます。たとえば、ユーザーが自分自身のグループを作成できるようにする場合、この割り当て量を使用すると、ユーザーが作成できるグループの合計数を制限できます。オブジェクト作成割り当て量のデフォルトは **unlimited** で、これはプリンシパルが作成できるオブジェクトの数に制限がないことを示します。値 0 (ゼロ) の場合は、プリンシパルはレジストリー・オブジェクトを作成することができません。

プリンシパルがレジストリー・オブジェクトを作成するたびに、プリンシパルのオブジェクト作成割り当て量は 1 つ減らされます。オブジェクト作成割り当て量が 0 になると、 **dcecp principal modify** コマンドでオブジェクト作成割り当て量を 0 以外の数にリセットしないかぎり、プリンシパルはレジストリー・オブジェクトの作成を禁止されます。プリンシパルの作成したオブジェクトが削除されても、プリンシパルのオブジェクト作成割り当て量は増加しない ことに注意してください。

プリンシパルの現在のオブジェクト作成割り当て量を表示するには、 **dcecp principal show** コマンドを使用します。このコマンドは、プリンシパルが現在、作成することを許されているオブジェクトの合計数 (つまり、元の割り当て量からプリンシパルが作成したオブジェクトの数を引いた数) を表示します。

汎用固有 ID と UNIX ID

DCE セキュリティー・サービスは、プリンシパル、グループ、またはオーガニゼーションの 1 次名を自動的に UUID と関連付けます。 UUID は、オブジェクト (つまり、 UNIX システムで UNIX 番号 (UNIX ID) が実行する機能) を識別します。(レジストリー・データベースにも UNIX 番号が格納されていますが、これらは、UNIX プログラムとの互換性のためだけに使用されます。)

通常、ユーザーは UUID について意識する必要はありません。それらは自動的に作成され、また保守されます。ただし、DCE セキュリティー・サービスは名前を出力し、ユーザーは名前によってオブジェクトにアクセスすることができますが、セキュリティー・サービスの内部ではオブジェクトをすべて UUID で識別していることに注意してください。プリンシパルをレジストリーから削除すると、プリンシパルの UUID も削除されます。プリンシパルが所有しているオブジェクト (ファイル、プログラム) はすべて、対応する名前を持たない UUID である、 ″オーファン″ (親のない) UUID に関連付けられます。これは、現在、削除されたプリンシパルによってオブジェクトが所有されていることを意味します。他のプリンシパルに対して、そのオブジェクトへのアクセスが以前に与えられなかった場合は、そのオブジェクトはアクセスできなくなります。

この問題を解決するためには、 **-uuid** オプションを指定した **dcecp principal create** コマンドを使用して UUID を名前と関連付け、親がなくなった (オーファンとなった) オブジェクトを受け入れます。 UUID は、DCE 制御プログラムの **principal create** コマンドを使用してオブジェクトが作成される時に、自動的に割り当てられます。したがって、単に新しいユーザーを追加して、これまで使用されていた UUID を獲得することはできません。獲得するためには、 **-uuid** オプションを指定して **dcecp principal create** コマンドを実行する必要があります。

レジストリー内の UNIX 番号は、レジストリー属性としてセットされた番号の範囲内でなければなりません。アカウントを作成または変更するためにコマンド行に UNIX 番号を入力するとき、 100 より小さい番号は避けてください。これらの番号は、通常、システム・アカウント用に予約されているからです。

プリンシパルの追加および保守

dcecp principal create コマンドを使用してプリンシパルを作成します。プリンシパルのアカウントを作成するには、その前にプリンシパルがすでに存在しなければなりません。 **dcecp principal create** コマンドを使用するとき、引き数としてプリンシパルの 1 次名を入力する必要があります。その他に、表18 に要約されている、属性オプションを指定することができます。

表 18. プリンシパルを作成するための属性オプション

オプション	意味
-fullname *namestring*	1 次名をさらに完全に記述するために使用するオプショナル名。スペースを含めるには、フルネームを中括弧で囲みます。デフォルトはブランクです。

表 18. プリンシパルを作成するための属性オプション (続き)

オプション	意味
-uid *integer*	プリンシパルに関連付けられる必須の UNIX ID。この番号は、ユーザーが明示的に入力するか、または自動的に生成することができます。これを入力する場合、入力する数字は **registry modify** コマンドでセットされた最大許容 UNIX 番号 (**maxuid** 属性) を超えることはできません。ただし、**registry modify** コマンドでプリンシパルにセットされた下限の UNIX 番号 (**minuid** 属性) より小さな数字を入力することはできます。番号が自動的に割り当てられるようにすると、番号は、下限の UNIX 番号と最大 UNIX 番号で定義された範囲内に入ります。
-quota *quota*	プリンシパルが作成できるレジストリー・オブジェクトの数であり、プリンシパルの*作成割り当て量*と呼ばれます。プリンシパルがレジストリー・オブジェクトを無制限に作成できるようにするには、割り当て量をセットしないテキスト・ストリング **unlimited** を入力します。プリンシパルがレジストリー・オブジェクトを作成できないようにするには、 0 を入力します。*quota* 引き数のデフォルトは **unlimited** です。

注: これらの標準のプリンシパル属性に加えて、 ERA インスタンスをプリンシパルに付加すれば、事前認証、パスワードの強さとパスワードの生成、および誤ったログインに対する処理などの、 DCE セキュリティーの側面を制御することができます。これらの予約済み ERA に関する情報については、 395ページの『第31章 アカウントの作成および保守』を参照してください。 ERA 全般については、 417ページの『第32章 拡張レジストリー属性の作成および使用』を参照してください。

プリンシパルの追加

プリンシパルをレジストリーに追加するには、 **principal create** コマンドを使用します。たとえば、次のサンプル・コマンドでは、 1 次名が **mahler** で、フルネームが **gustav mahler** のプリンシパルを作成します。

```
dcecp> principal create mahler -fullname {gustav mahler} -quota 5
dcecp>
```

この例では、UNIX 番号はデフォルトで自動的に生成される番号となります。プリンシパルに割り当てられるフルネーム (**gustav mahler**) にスペースが含まれているために、フルネームが中括弧で囲まれていることに注意してください。

1 つの **principal create** コマンドで、複数のプリンシパルを作成できます。これを行うには、プリンシパル名を中括弧で囲み、間をスペースで区切ります。たとえば、プリンシパル **bach**、**britten**、**mahler**、および **satie** を作成するには、次のように入力します。

```
dcecp> principal create {bach britten mahler satie}
dcecp>
```

複数のプリンシパルを作成する場合、プリンシパルの UNIX ID はデフォルトのシステム割り当ての ID にする必要があります。これは、コマンド行に属性オプショ

ンを含めると、その属性値が各プリンシパルに割り当てられるためです。たとえば、次のサンプル・コマンドは、プリンシパル **bach**、**britten**、**mahler** を作成し、それぞれに 5 のオブジェクト作成割り当て量を割り当てます。

```
dcecp> principal create {bach britten mahler satie} -quota 5
dcecp>
```

プリンシパルの変更

プリンシパルの 1 次名とプリンシパルに関するその他の情報は変更することができます。さらに、1 次名を別名に、別名を 1 次名に変えることができます。 1 次名を別名に変更すると同時に、別名を 1 次名には変更しない場合、操作によって名前が返される時に、別名の中から 1 つがランダムに選択されます。

1 次名の変更

1 次名を変更するには、**dcecp principal rename** コマンドを使用します。コマンドは次の形式で入力します。

principal rename *old_name* **-to** *new_name*

ここで、

old_name
変更するプリンシパルの 1 次名。

new_name
プリンシパルの新しい 1 次名。

以下は、**principal rename** コマンドを使用してフルネーム **mahlar** を **mahler** に変更する例を示しています。

```
dcecp> principal rename mahlar -to mahler
dcecp>
```

1 次名を変更すると、その変更は、プリンシパルがメンバーとなっている、すべてのグループとオーガニゼーションのメンバーシップ・リストにも反映されることに注意してください。

ホストが DCE セルに記録している間にホストのプリンシパル名を変更するというまれなケースでは、追加的なステップを実行して新しいプリンシパル名でホストの証明書を更新しない限り、既存のホストの証明書は無効になります。

ホストの証明書は、 DCE ホスト上でセキュリティー・クライアント機能を実行する **secval** プロセスによって管理されます。通常、ホストのスタート直後に **secval** プロセスは、ホストを DCE セルにログインさせ、ホストの証明書を入手し、それらをホストに格納します。プリンシパルの名前を変更したあと、 **secval** プロセスを非活動状態にしてから、再び活動状態にして、これらの証明書を更新します。次の例は、リモート・ホスト **persephone** でのこれらの操作を示しています。

```
dcecp> secval deactivate /.:/hosts/persephone/config/secval
dcecp> secval activate /.:/hosts/persephone/config/secval
dcecp>
```

注: このリリースの DCE では、アグリゲートのプリンシパル、グループ、またはオーガニゼーション (pgo) を移動してそれ自身の下位にするような操作はサポートされません。たとえば、以下のような一連のレジストリー操作は使用できません。

```
dcecp -c group create foo/bar
dcecp -c group create foo
dcecp -c group rename foo -to foo/xyz
```

プリンシパル情報の変更

プリンシパルの情報 (ただし、UNIX ID とユーザー ID を除く) を変更するには、**dcecp principal modify** コマンドを使用します。次の例では、**principal modify** コマンドを使用して、プリンシパル **mahler** のオブジェクト割り当て量を 10 に変更します。

```
dcecp> principal modify mahlar -quota 10
dcecp>
```

プリンシパルおよび別名の削除

プリンシパルまたは別名を削除すると、システムは自動的にそのプリンシパルまたは別名のアカウントを削除します。たとえば、プリンシパル **mahler** を削除すると、 **mahler composers classic** アカウントも削除されます。プリンシパルの別名 **gustav** を削除すると、 **gustav music_admin classic** アカウントも削除されます。グループ別名 **music_admin** を削除すると、 **gustav music_admin classic** アカウントも削除されます。プリンシパルまたはプリンシパルの別名を削除すると、プリンシパル /UUID が所有しているオブジェクトはオーファン・オブジェクトになることに注意してください。

次の例は、**principal delete** コマンドを使用して、 **mahler** というプリンシパルを削除します。

```
dcecp> principal delete mahler
dcecp>
```

1 つの **principal delete** コマンドで、複数のプリンシパルまたは別名を削除することができます。これを行うには、プリンシパル名を中括弧で囲み、間をスペースで区切ります。たとえば、プリンシパル **bach**、**britten**、および **mahler** を削除するには、次のように入力します。

```
dcecp> principal delete {bach britten mahler}
dcecp>
```

プリンシパルの拡張セキュリティー属性

ERA インスタンスをプリンシパルに付加して、 DCE ログインおよびパスワード・セキュリティーのいくつかの側面を管理することができます。 ERA を使用すると、次のような制御が可能です。

- プリンシパルのログイン要求に必要な、認証セキュリティーのレベル
- 無効なログインに対する処理
- プリンシパルのパスワードの強さとプリンシパルのパスワードの生成
- 期限切れのパスワードを使ったプリンシパルによるログイン試行に対する処理

これらの ERA については、この後のセクションで紹介し説明します。 **dcecp** を使用してこれらの ERA をプリンシパルに付加する方法については、 417ページの『第32章 拡張レジストリー属性の作成および使用』を参照してください。

DCE 認証

DCE バージョン 1.1 より以前のバージョンで DCE 認証プロトコルの基礎として使用されていた、 Kerberos V5 認証プロトコルのある種のセキュリティー上の欠陥に、認証は対応しています。これらの欠陥は、次のことから生じます。

- セキュリティー・サーバーは、ユーザーがパスワードを知っているかどうかを検証せずに、クライアントのログイン要求に応答する
- 非常に弱いユーザー・パスワードを使用してプレーン・テキスト・データを暗号化し、それをネットワークに送信する

これらの方法は、侵入を受けやすくしています。侵入者はネットワーク伝送を入手し、続いてオフラインで侵入して、ユーザーのパスワードを入手します。この種の攻撃は、成功した場合、 DCE セル (およびセルと信頼関係にある他のすべてのセル) のセキュリティーを損なう可能性があります。

DCE 認証では、以下の対策によって、このような侵入が成功する可能性を小さくしています。

- ログイン要求を行うプリンシパルの事前認証 (つまり、 DCE セキュリティー・サービスが要求者の ID を検査してから要求に応答する)
- セキュリティー・クライアントとサーバーの間の妥当性検査を含む、すべてのネットワーク伝送を暗号化する強いキーの使用

最も安全なものからあまり安全でないものまで、 5 つの認証レベルがあります。下のレベルになるほど、事前認証プロトコルはより緩やかです。 *pre_auth_req* ERA (これについては、次のセクションで説明) のインスタンスをプリンシパルに付加することにより、管理者は、セキュリティー・サーバーがプリンシパルを認証する際に受け入れる、最低レベルの事前認証を制御することができます。

事前認証プロトコルは、次のとおりです。

- パブリック・キー*証明書* プロトコル。これを利用すると、DCE ユーザーは、X.509v3 証明書およびそれに関連したパブリック・キー・ペアを使って、サード・パーティー・パブリック・キーに基づいて識別を証明できます。パブリック・キー証明書ログインを使用可能にする方法については、 367ページの『パブリック・キー証明書ログイン』を参照してください。
- パブリック・キー・プロトコル。これを利用すると、DCE ユーザーは、 DCE レジストリーが生成して保管するパブリック・キー・ペアを使って識別を証明できます。このプロトコルに代わって、パブリック・キー証明書プロトコルがサポートされています。詳細については、 365ページの『パブリック・キー証明書プロトコルを使用可能にする』を参照してください。
- サード・パーティー・プロトコル。これは高レベルのセキュリティーを提供します。特別な理由がない限り、これより低いレベルの事前認証をプリンシパルに指定しないでください。 (次の黒丸で示された項目にある、 **cell_admin** に関する注を参照してください。) DCE バージョン 1.1 クライアントは、常にこのプロトコルを使用して認証要求を作成します。ただし、 (たとえば、セルの始動時、ま

たは **secval** プロセスがダウンしているなど) サード・パーティー要求を作成するために必要なマシン・セッション・キーが使用可能でない場合は、認証要求を作成できません。

- タイム・スタンプ・プロトコル。これは、中間のレベルのセキュリティーを提供します。タイム・スタンプ事前認証は、クライアントが上記のサード・パーティー認証要求を作成できないときに、プリンシパル (セル管理者や非対話式プリンシパルなど) が操作を行わなければならない場合にだけ指定してください。

  これらの場合、クライアントはタイム・スタンプ・ログイン要求を構成して転送します。

  特にセル管理者は、タイム・スタンプ・ログイン能力を持っていなければなりません。その理由は、**cell_admin** は、セルの初期構成のときに、初期マシン・キーをセットアップするためにログインする必要があるからです。

- DCE バージョン 1.0 (Kerberos V5) プロトコル。これは、DCE バージョン 1.1 以前のクライアントを認証するためだけに使用され、事前認証セキュリティーは提供しません。

パブリック・キー証明書プロトコルを使用可能にする

注: DCE for AIX および Solaris クライアントは、 OSF DCE 1.2.2 パブリック・キー・プロトコルを使用した認証をサポートしなくなりました。このプロトコルに代わって、パブリック・キー証明書プロトコルがサポートされています。

DCE for AIX および Solaris クライアントでは、 OSF DCE 1.2.2 パブリック・キーに関連した **dcecp** コマンドを使用して、管理者がパブリック・キー・ユーザーの管理を行うことはできません。しかし、ユーザーのプライベート・キーを収めているファイルのフォーマットは、プラットホームで固有です。このファイルが使用されるシステムと同じタイプのシステムにおいて、これを生成しなければなりません。

DCE for AIX および Solaris セキュリティー・サーバーは、 OSF DCE 1.2.2 パブリック・キー・プロトコルをサポートしません。このセキュリティー・サーバーは、 OSF DCE 1.2.2 に基づいたクライアントからのパブリック・キー要求を処理し、この機能をサポートします。

デフォルトでは、パブリック・キー・ログイン認証は使用不可にされています。これを使用可能にし、ユーザー用に構成するには、次のステップに従います。

1. パブリック・キー・ソフトウェアが、マスター DCE セキュリティー・サーバー上とレプリカ・サーバー上にインストールされていることを確認します。パブリック・キー・ソフトウェアは DCE バージョン 1.2.2 およびこれ以降で使用可能です。
2. セルのパブリック・キー環境を初期化します。セルを DCE 1.1 から DCE 1.2.2 に移行する場合は、 2a および 366ページの2b のステップに従います。その他の場合は、366ページの2b のステップに従います。
   a. 次のようにして、マスター上のパブリック・キーを使用可能にします。

      ```
      dcecp> registry modify -version secd.dce.1.2.2
      dcecp>
      ```

マスターは、パブリック・キー・ソフトウェアがインストールされている各レプリカにバージョン情報を伝搬します。パブリック・キー・ソフトウェアがインストールされていないすべてのレプリカは、自動的にシャットダウンされます。

b. **cell_admin** として、ローカル・セルの **krbtgt** プリンシパルのアカウント用に、初期キーのペア (パブリック・キーとプライベート・キー) を生成するよう要求します。

```
dcecp> account modify krbtgt/cell-name -pkgenprivkey  module size¥
> -mypwd pwd
dcecp>
```

レジストリーは、キーのペアを生成し、それを保管します。キーのペアのパブリック・キーの部分は、セルの **krbtgt** プリンシパルに付加された、**DCEPKAuthentication** ERA に保管されます。プライベート・キーの部分は、既存のパスワード保護暗号化のもとで、レジストリー内に保管されます。

このリリースでは、 512 から 1024 までの範囲のモジュラス・サイズ をサポートしています。通常のユーザー用のパブリック・キー・ペア生成に使用するモジュラスの最小サイズは 512 ですが、キー・ペア生成の推奨モジュラス・サイズは 768 ビットです。長期間実行するアプリケーションに対しては、 1024 ビットのモジュラス・サイズを推奨します。

このコマンドを使用して、ローカル・セルの **krbtgt** プリンシパルの既存のキーのペアを変更することもできることに注意してください。セルのキー・ペア **krbtgt**/*cell-name* 用にサポートされているモジュラス・サイズは、デフォルト値の 1024 のみです。

3. セル管理者のワークステーション上で、ユーザーのパブリック・キー・アカウントを構成します。

```
dcecp> account create name ¥
> -group group name ¥
> -mypwd cell-admin-password ¥
> -organization org_name ¥
> -password user-password ¥
> -pkmechanism file ¥
> -pkkeycipherusage {{generatekey modulus-size} {newpassphrase string}} ¥
> -pksignatureusage {{generatekey modulus-size} {newpassphrase string}}
```

このコマンドは、パブリック・キー・アカウントに対して以下のことを行います。

- プライベート・キー情報をファイルに保管するよう指示する。
- 新しいパブリック・キーのペアをランダムに生成し、ユーザーが暗号化目的のために提供する、新しいパスワード句をこれに付加する。キーのペアのパブリック・キーの部分は ERA に保管され、プライベート・キーの部分はファイルに保管されます。
- 新しい符号つきキーのペアをランダムに生成し、ユーザーが提供する新しいパスワード句をこれに付加する。パブリック・キーの部分は ERA に保管され、プライベート・キーはファイルに保管されます。

4. パブリック・キー・メカニズム・ファイルを、セル管理者のワークステーションからユーザーのワークステーションに移し、所有者をユーザーに変更する。新し

いパスワード句をユーザーに提供します。なお、パスワード句はユーザー以外の誰にも分からないものに変更するよう、ユーザーに強く勧めてください。パスワード句を変更するためには、ユーザーは **dcecp account modify** コマンドを呼び出します。

```
dcecp> account modify name ¥
> -pkkeycipherusage {{oldpassphrase string} {newpassphrase newstring}} ¥
> -pksignatureusage {{oldpassphrase string} {newpassphrase newstring}}
```

デフォルトでは、セル管理者だけがパブリック・キー・アカウントを作成、または変更できます。しかし、セル管理者は、パブリック・キー ERA の ACL マネージャーを変更して、ユーザーが自分自身のキーのペアを変更できるようにすることができます。 ACL マネージャーを変更するには、以下のようにします。

```
dcecp> xattrschema modify /.:/sec/xattrschema/DCEPKAuthentication ¥
> -aclmgr {principal {query r} {update u} {test r} {delete m}}
dcecp> xattrschema modify /.:/sec/xattrschema/DCEPKKeyEncipherment ¥
> -aclmgr {principal {query r} {update u} {test r} {delete m}}
```

パブリック・キー証明書ログイン

DCE セキュリティー・クライアントは、パブリック・キー事前認証プロトコルを使用して、ユーザー用のチケット発行チケット (TGT) を獲得します。 OSF-RFC 68.4 は DCE 1.2.2 パブリック・キー・プロトコルを拡張して、 DCE ユーザーが X.509v3 ディジタル証明書を使って DCE 認証サービスに対して識別を証明できるようにします。以下の情報は、DCE 3.2 for AIX および Solaris に実装されている、 OSF-RFC 68.4 に基づいたパブリック・キー証明書ログインについて説明しています。これを実装するには、Entrust Public Key Infrastructure (PKI) が必要です。

パブリック・キー・ログインの概説: DCE for AIX および Solaris は、 DCE ユーザーが共用の機密キー・パスワードではなく、 X.509v3 ディジタル証明書およびそれに関連したパブリック・キー・ペアを使用して、 DCE 認証サービスに識別を証明できるようにします。このような認証メカニズムの 1 つの利点は、 DCE セキュリティー・サーバーが侵入を受けた場合、パブリック・キー・ユーザーの識別情報が侵入者にわからないという点です。共用の機密キー認証の場合は、すべてのユーザーの機密キーが侵入者に知られてしまう可能性があります。もう 1 つの明確な利点は、パブリック・キー暗号化の方式によって、基本的な認証の流れがより安全になることです。

現在 Entrust PKI を使用していて、認証用に Entrust ユーザーを DCE ユーザーにマップして DCE の提供するリソースにアクセスする必要の生じたお客さまのために、このような機能強化が行われました。この変更は、初期 TGT の獲得にのみ適用されます。いったん TGT を受信すると、追加のサービス・チケットは以前の方法で獲得されます。

DCE クライアントと DCE セキュリティー・サーバーの間で交換される情報の変更に加えて、この機能強化には新しい Identity Mapping Server (IDMS) が含まれています。このサーバーは DCE セキュリティー・サーバーに呼び出されて、ユーザーのディジタル証明書を DCE プリンシパル名にマップします。 368ページの図53 は、この機能強化によって確立された認証の流れを示しています。

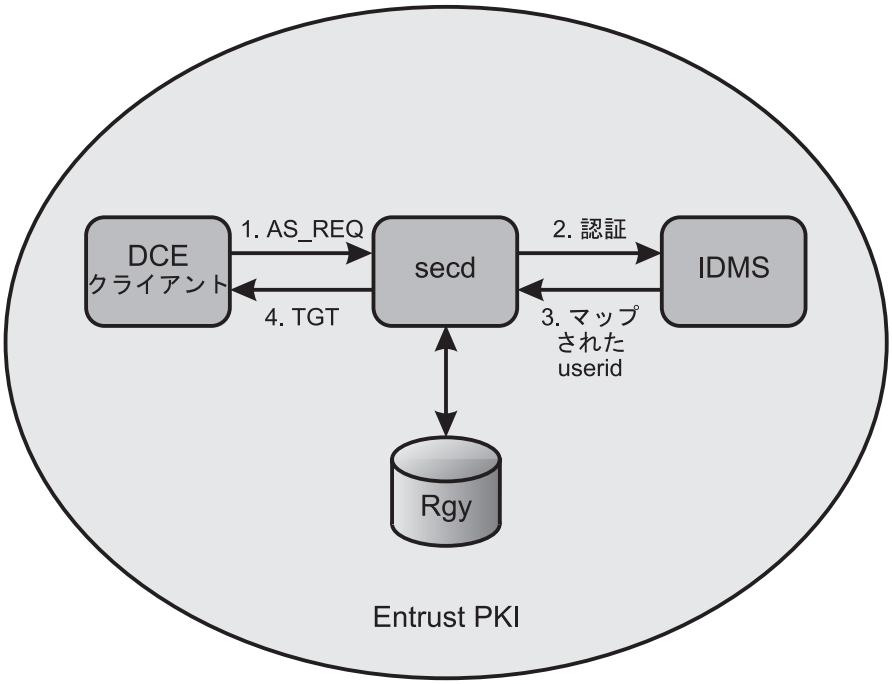


図 53. パブリック・キー証明書の認証の流れ

パブリック・キー証明書ログインの使用のための Entrust 前提条件: この拡張機能には Entrust PKI が必要です。さらに、DCE クライアントおよびセキュリティー・サーバーに Entrust クライアントがインストールされていなければなりません。

この拡張機能のそれぞれのユーザーごとに、 Entrust 管理者は Entrust ユーザーを作成して、各ユーザーに対して署名用および暗号化用のパブリック・キー・ペアを発行する必要があります。それらのパブリック・キーは Entrust 認証局 (CA) によってディジタル署名されるパブリック・キー証明書に保管され、 Entrust ユーザー・プロファイルに保管されます。プライベート・キーは Entrust ユーザーのプロファイルに保管されて、パスフレーズ (Entrust 資料ではこれをパスワードと呼ぶ) によって保護されます。あるユーザーに関連した Entrust プロファイルは、そのユーザーのクライアント・システム上に収める必要があります。

さらに、それぞれの Entrust ユーザーに関連した DCE プリンシパルを、ユーザーの X.500 ディレクトリー・エントリーの属性として追加する必要があります。 Entrust は X.500 ディレクトリーを必要とし、ユーザーの X.500 識別名 (DN) によってユーザーを追跡します。 Entrust ユーザーを作成および管理するためのツールは Entrust によって提供されています。それらは既存の DCE ツールには統合されていません。 Entrust ユーザーの作成に関する詳細情報は、 Entrust PKI に付属の資料を参照してください。

また、Entrust 管理者は、 DCE セキュリティー・サーバーが使用するための Entrust ユーザーを作成する必要があります。セル内のすべてのセキュリティー・サーバーが、この Entrust ユーザー用の証明書および対応するパブリック・キー・ペアを使うようにすることもできます。また、各 DCE セキュリティー・サーバーごとに個別の Entrust ユーザーを作成することもできます。 Entrust ユーザー・プロファイルは、それぞれの DCE セキュリティー・サーバー上に保管する必要があります。

IDMS は、そのシステム上でセキュリティー・サーバーが使用している Entrust ユーザー・プロファイルと同じものを使用します。

パブリック・キー証明書ログインを使用可能にする: パブリック・キー証明書ログインを使用可能にするために、 DCE セキュリティー・サーバーに必要な 2 つの変更を行いました。第一に、DCE 構成ツールを更新し、IDMS の追加をサポートして、 DCE セキュリティー・サーバーの使用する Entrust ユーザー・プロファイルの場所を指定できるようにしました。第二に、DCE レジストリーの新しいバージョン 1.2.2a を作成しました。

パブリック・キー証明書ログインが DCE セルで使用できるのは、そのセルのマスター・セキュリティー・サーバーをバージョン 1.2.2a で実行している場合に限ります。 DCE 3.2 for AIX および Solaris を使用して構成したセルのデフォルトは、バージョン 1.2.2 です。 DCE の以前のバージョンから移行されたセルは、移行前に実行していたセキュリティー・サーバーと同じバージョンを引き続き実行します。

レジストリー・バージョン番号を変更してパブリック・キー証明書ログインを使用可能にするには、 **dcecp registry modify** コマンドを使用してください。

```
dcecp> registry modify -version secd.dce.1.2.2a
dcecp>
```

警告: このコマンドを実行する時に、セル内のいずれかのセキュリティー・レプリカが、セキュリティー・サーバー・バージョン 1.2.2a をサポートしない DCE のバージョンを実行している場合は、レプリカが自動的にシャットダウンされます。

既存の DCE セルを DCE 3.2 for AIX および Solaris に移行するための詳細は、*IBM DCE for AIX バージョン 3.2: 概説とインストール AIX 版* または *IBM DCE for Solaris バージョン 3.2: 概説とインストール Solaris 版* の移行に関するセクションを参照してください。

パブリック・キー証明書ログインを使用する 1 つの DCE セルには、少なくとも 1 つの IDMS が必要です。セル内の各セキュリティー・サーバーごとに 1 つの IDMS を構成するのが理想的です。以下のリストは、コマンド行構成オプションに関する詳細です。これらのオプションは **SMIT** でもサポートされており、 DCE セキュリティー・サーバーまたは識別マッピング・サーバーの構成時に使用できます。

- (マスターまたはレプリカの) セキュリティー・サーバーの構成時にパブリック・キー証明書ログインを使用可能にして、 IDMS をマシンで使用するには、 (適切な) **sec_srv or sec_rep** コンポーネント、および **idms_srv** コンポーネントを指定した上で、以下のようなオプションを含めてください。

  -certificate_based_login yes

  -kdc_ini_file *kdc_ini_file*

  -kdc_profile *kdc_profile*

  -kdc_passphrase *kdc_passphrase*
- パブリック・キー証明書ログインを使用可能にして、 IDMS を使わずにセキュリティー・サーバーを構成するには、上記の例で **idms_srv** コンポーネント・オプションを省略します。

- すでに構成されたセキュリティー・サーバーでパブリック・キー証明書ログインを使用可能にするには、以下のオプションを指定してください。

  -certificate_based_login yes

  -kdc_ini_file *kdc_ini_file*

  -kdc_profile *kdc_profile*

  -kdc_passphrase *kdc_passphrase*

  セキュリティー・サーバーのバージョンが 1.2.2a であれば、次に **secd** をスタートする時に、パブリック・キー証明書ログインが使用可能になります。
- すでに構成された、パブリック・キー証明書ログインが使用可能になっていないセキュリティー・サーバーに IDMS を追加するには、 **idms_srv** コンポーネント・オプションとともに以下のオプションを指定してください。

  -certificate_based_login yes

  -kdc_ini_file *kdc_ini_file*

  -kdc_profile *kdc_profile*

  -kdc_passphrase *kdc_passphrase*
- すでに構成された、パブリック・キー証明書ログインが使用可能になっているセキュリティー・サーバーに IDMS を追加するには、単に **idms_srv** コンポーネント・オプションだけを指定します。

これらの構成オプションについて、詳しくは *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

パブリック・キー証明書ログインを使用した認証: DCE ログイン・インターフェースは変更されていませんが、入力値の意味は一部変更されています。パブリック・キー証明書ログインを利用すると、ユーザーの DCE プリンシパル名ではなく、 Entrust ユーザー・プロファイルを提供することによって、ログインして識別を証明することができます。 DCE パスワードの代わりに、 Entrust ユーザー・プロファイルに保管されているプライベート・キーをアンロックするためのパスフレーズを指定する必要があります。

この機能強化によって変更された DCE API は、 **sec_login_validate_identity( )** および **sec_login_valid_and_cert_ident( )** です。これらの API には追加のフラグや引き数がなくなりました。その代わりに、既存の引き数の意味の解釈が変わりました。これによって、既存のログイン・ユーティリティーおよびプラットホーム固有の統合ログイン・メカニズムが、再書き込みや再コンパイルを行わずに、Entrust パブリック・キー証明書を使って DCE に認証を行えるようになりました。

たとえば、ユーザーの Entrust プロファイルが **henry.epf**、パスフレーズが **Rottweiler9** であれば、以下のコマンドを使用して DCE にログインできます。

```
dce_login henry Rottweiler9
```

ユーザーの Entrust プロファイルとパスフレーズ、および Entrust クライアントの **entrust.ini** ファイルによって、 DCE のルーチンが Entrust **ETLogin( )** API を呼び出すことができます。これに基づいて、Entrust によるその後のパブリック・キーの署名、検査、暗号化、および暗号復号化の操作が行われます。これらの操作は、DCE にログインする際にユーザー妥当性検査に使用する事前認証データを構築および処理するためのものです。

ユーザーの Entrust プロファイルとパスフレーズは、先に説明したセキュリティー・ログイン API に引き数として渡されます。 **entrust.ini** ファイルへの全パスは、 **.entrustrc** ファイルから得られます。このファイルは、 Entrust クライアントの構成時にホーム・ディレクトリーに作成されます。このファイルの中の ENTRUSTDIR エントリーが、 **entrust.ini** ファイルへの全パス名を指定しています。 Entrust プロファイルへの全パス名は、 **.entrustrc** ファイルの中で、最後に使われたプロファイルを検索して得られます。この中にプロファイルが見付からない場合は、 **entrust.ini** ファイルで指定された *DefaultProfileLocation* が検索されます。

以前の認証への復帰: パブリック・キー証明書ログインが失敗した場合、ログイン・インターフェースの構造を、 (以前の DCE パスワードを使用した) 共用の機密キーに復帰させることができます。これは、 DCE セキュリティー・クライアントが DCE セキュリティー・サーバーに認証を要求する時、 DCE セキュリティー・クライアント・ランタイムが、パブリック・キー証明書ログイン・プロトコル用、および DCE サード・パーティー・プロトコルまたは DCE タイム・スタンプのいずれかのための、事前認証データを構築して送信するためです。 DCE プリンシパル名を使って Entrust ユーザー・プロファイル名を付け、 Entrust パスフレーズが DCE パスワードと一致する場合は、この復帰は可視になります。復帰が可能かどうかを判別するには、 *pre_auth_req* ERA を使用します。

識別マッピング・サーバー: 識別マッピング・サーバー (IDMS) は、ユーザーのパブリック・キー証明書を DCE プリンシパル名にマップする新しい RPC サーバーです。パブリック・キー証明書ログインを使用する 1 つの DCE セルには、少なくとも 1 つの IDMS が必要です。パブリック・キー証明書ログインを使用する DCE セル内の各セキュリティー・サーバーごとに、 1 つの IDMS を構成するのが理想的です。ユーザーが認証のためにパブリック・キー証明書を使って DCE にログインした時、 DCE セキュリティー・サーバーは IDMS を呼び出します。 IDMS では、従来の 1 対 1 のマッピングに加えて、多数の Entrust ユーザーを単一の DCE ユーザーにマップできます。

IDMS では、ある Entrust ユーザーに関連した DCE プリンシパルを、そのユーザーの X.500 ディレクトリー・エントリーの追加属性として追加する必要があります。 Entrust は X.500 ディレクトリーを必要とし、ユーザーの X.500 DN によってユーザーを追跡します。特に、Entrust ユーザーを表すオブジェクト用に、 **dcePrincipal** という属性名をディレクトリー・スキーマに追加しなければなりません。通常、Entrust ユーザーは、タイプ **organizationalPerson** またはタイプ **entrustUser**、あるいはその両方です。その後、ユーザーの証明書の中の X.500 DN を使用し、ディレクトリー内でユーザーの **dcePrincipal** 属性を検索することによって、 Entrust ユーザーから DCE プリンシパルへのマッピングが行われます。属性の追加についての追加情報は、 X.500 ディレクトリー・サービスに付属の文書を参照してください。

識別マッピングがこのような方法で実行されるのは、 DCE セキュリティー・サーバーが別個のサーバーを呼び出してマッピングを獲得する方が、より柔軟性に富んでいるためです。これによって、マッピング・アルゴリズムのカスタマイズが可能になります。個々の必要に応じて識別マッピング・ポリシーが異なるため、デフォ

ルト IDMS のソース・コードが DCE サンプル・プログラムとして提供されています。このサンプル・プログラムは、 **/usr/lpp/dce/examples/idms** ディレクトリーにインストールされています。

パブリック・キー証明書ログインの制限: パブリック・キー認証を使用するよう構成したアカウントには、いくつかの制限があります。これらは以下のとおりです。

- 予約済み DCE プリンシパルは、パブリック・キー証明書ログインを使用できません。 DCE 3.2 for AIX および Solaris では、予約済みフラグ・セットを持つプリンシパルには次のものがあります。
  - セルを最初に構成する時に指定する管理 ID。
  - セルの krbtgt プリンシパル。
  - システムの自己プリンシパル、つまり初期マスター・セキュリティー・サーバー。

  この制限による副次作用の 1 つは、 DCE 構成中に Entrust ユーザー・プロファイルの名前を cell_admin ID として提供できないことです。
- DCE パスワードを提供しない限り、 **kinit** コマンドを使用して、有効期限切れの DCE 証明書を更新することはできません。この更新操作のための、 Entrust ユーザー・プロファイルとパスフレーズの使用はサポートされていません。 Entrust ユーザー・プロファイル名およびパスフレーズが DCE プリンシパル名およびパスワードと同期化している場合は、この制限がユーザーにとって可視になります。
- 複数の Entrust ユーザーを単一の DCE プリンシパルにマップする時、監査やアクセス制御などの DCE 機能性の詳細レベルは低下します。監査レコードおよびアクセス制御のチェックには、 DCE プリンシパル情報のみが使用されます。
- パスワードの強さチェックが必要な *pwd_val_type* ERA が DCE プリンシパルに付加されている場合、これらのチェックはそのプリンシパル用の DCE パスワードに対してのみ実施されます。 Entrust PKI は、 Entrust パスフレーズ上で実施される規則の別個のセットを確立します。
- キー管理 API は、共用の機密キー認証プロトコルを使用するアプリケーションのみが使います。パブリック・キー証明書ログインを使用するアプリケーションは、ユーザー対ユーザーのプロトコルを使う必要があります。
- GSSAPI を使用する際、DCE 管理者は、 DCE レジストリー・データベース内にイニシエーターおよびアクセプター用のアカウントを 1 つ設定しなければなりません。アクセプターは、パブリック・キー認証を使用できません。イニシエーター用のアカウントについては、適用される制限はありません。
- DCE 3.2 for AIX および Solaris の PKC ログイン・サポートは、 IETF RFC 1510 の 1998 年初期のドラフト・バージョンである、 Public Key Cryptography for Initial Authentication in Kerberos (PKINIT) に基づいています。このドラフトは完成していないため、 PKINIT の将来のバージョンとの相互運用性は保証されません。
- DCE パスワードを提供しない限り、 **dce_login -r** コマンドを使用して、有効期限切れの DCE 証明書を更新することはできません。この更新操作のための、 Entrust ユーザー・プロファイルとパスフレーズの使用はサポートされていません。

- **(AIX のみ)** AIX/DCE 統合ログイン・オプションを使用した PKC ログインは、現在のところサポートされていません。

Entrust の要件および制限: Entrust 製品は、パブリック・キー証明書ログインの使用を計画している場合にのみ必要です。 AIX および Solaris では、この機能を使用するための以下のような要件があります。

- Entrust/Toolkit バージョン 5.0.2 の、 Entrust Engine ライブラリー (PKC ログインの使用が可能なそれぞれの DCE クライアント上に)
- Entrust/Toolkit バージョン 5.0.2 の、 Entrust Engine ライブラリー (それぞれの DCE セキュリティー・サーバーおよび識別マッピング・サーバー上に)

Entrust/Toolkit バージョン 5.0.2 には、パブリック・キー証明書ログイン機能のテスト中に生じた問題を修正するパッチが含まれています。 Entrust/Toolkit のこのパッチ・レベルは、 Entrust Technologies 社の FTP サイトから入手できます。

Entrust Public Key Infrastructure は、 DCE システムには必要ではありませんが、ユーザーへの証明書の発行のために使用する必要があります。 Entrust/PKI の推奨レベルはバージョン 5.0 です。

PKC (パブリック・キー証明書) ログインのテスト中に、以下のような Entrust/Toolkit の制限が発見されました。

- かなりの数の同時ログイン・アクティビティーが発生する環境では、認証メッセージの処理中に、 DCE セキュリティー・サーバー Entrust からエラーを受け取ることがあります。その結果として、PKC ログイン試行が失敗します。 **secd** スタート前に、 DCE 保守容易性ルーティング・ファイルで NOTICE レベルのログ記録が使用可能になっている場合、 DCE セキュリティー・サーバーは、 Entrust エラーをファイル **notice.log** に記録します。
- Entrust/Lite を使用した PKC ログイン、およびライト・モード での Entrust/PKI を使用した PKC ログインはサポートされていません。

パブリック・キー証明書ログインを使用するために必要なステップの要約: 以下のリストは、パブリック・キー証明書ログインの拡張機能を使用するために必要なステップを要約しています。

1. Entrust Public Key Infrastructure を構成して、 Entrust ユーザーを作成する。
2. DCE クライアントまたはセキュリティー・サーバーとなっている (あるいはそうする予定の) システム上に、 Entrust クライアントをインストールする。
3. Entrust ユーザー・プロファイルを、それぞれ DCE クライアントおよびセキュリティー・サーバーのシステム上に保管する。
4. DCE セルをレジストリー・バージョン 1.2.2a に構成または移行する。
5. IDMS およびパブリック・キー証明書オプションを、セル内の各セキュリティー・サーバーに構成する。
6. **dcePrincipal** 属性をユーザーの X.500 ディレクトリー・エントリーに追加する。
7. ユーザーは Entrust プロファイル名、およびプロファイルを保護するパスフレーズを指定して、ログインすることができる。

DCE ユーザー認証の管理

特定の DCE ユーザーの事前認証の管理を行うには、ユーザーのプリンシパルに *pre_auth_req* ERA のインスタンスを付加し、さらに DCE セキュリティー・サービスがプリンシパル用に受け入れるべき*最低のレベルのプロトコル* を示す値を、次のように指定します。

0 **(NONE)**。 DCE セキュリティー・サービスが、 4 つのプロトコル (DCE バージョン 1.1 より以前のプロトコルを含む) のいずれかを使用するログイン要求を、このプリンシパルから受け入れるべきであることを指定します。これは最も安全性の低いレベルであり、 DCE バージョン 1.1 サーバーが DCE バージョン 1.1 より以前のクライアントからのログイン要求を受け入れられるようにする目的でのみ用意されています。このレベルは、前述のように、侵入に最も弱いレベルです。

警告: プリンシパルに *pre_auth_req* ERA のインスタンスを付加することに失敗すると、 **0 (NONE)** を指定したのと同じことになります。

1 **(PADATA-ENC-TIMESTAMPS)**。これは DCE セキュリティー・サービスが、タイム・スタンプ、サード・パーティー、またはパブリック・キーのプロトコルを使用したログイン要求を、このプリンシパルから受け入れるべきことを指定します。タイム・スタンプ・プロトコルは、侵入者がセキュリティー・クライアントを装い、 DCE 認証サービスからの応答を攻撃することから保護します。ただし、このプロトコルは、ネットワークをモニターできるプロセスからの攻撃に対しては保護されていません。

2 **(PADATA-ENC-THIRD-PARTY)**。これは、DCE セキュリティー・サービスがこのプリンシパルから受け入れるログイン要求は、サード・パーティー・プロトコル、またはパブリック・キー・プロトコルを使用したログイン要求だけであることを指定します。このプロトコルは高レベルの DCE 事前認証を提供するとともに、侵入に対する保護を提供します。サード・パーティー事前認証では、ネットワークを通して送られる認証データはすべて、ローカル・マシンのプリンシパルと DCE セキュリティー・サービスだけが知っている、「強い」ランダム・キーを使用して暗号化されます。

3 **(PADATA-ENC-PUBLIC-KEY)**。これは、DCE セキュリティー・サービスがこのプリンシパルから受け付けるログイン要求が、パブリック・キー・プロトコルを使用したログイン要求だけであることを指定します。

4 **(PADATA-ENC-PUBLIC-KEY-CERTIFICATE)**。これは、DCE セキュリティー・サービスがこのプリンシパルから受け入れるログイン要求は、パブリック・キー証明書ログイン・プロトコルを使用したログイン要求だけであることを指定します。

DCE 認証サービスは、プリンシパルのログイン要求を受け取ると、常に要求と同じプロトコルを使用して応答しようとします。ただし、そのプリンシパルの *pre_auth_req* ERA 値によって、それが『禁じられていない』場合に限ります。 375 ページの表19 は、ログイン (認証) 要求タイプと *pre_auth_req* ERA 値のさまざまな組み合わせのもとで認証サービスが取るアクションを記述した、マトリックスです。

(事前認証プロトコルの操作を含む) DCE 認証についての詳細は、 *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* を参照してください。

次の **dcecp** コマンドの例では、プリンシパルを変更し、パブリック・キー証明書ログインが必要であることを指定する *pre_auth_req* ERA を付加します。

```
dcecp> principal modify name -add {pre_auth_req 2}
dcecp>
```

ユーザーがパブリック・キー証明書ログインの使用を許可されているものの、その使用が必須でない場合は、 *pre_auth_req* ERA の設定は必ずしも必要ではありません。 **dcecp** を使用してプリンシパルに ERA を付加する方法の詳細については、417ページの『第32章 拡張レジストリー属性の作成および使用』を参照してください。

Entrust メッセージ

Entrust Public Key Infrastructure が提供するメッセージは、 DCE がサポートするすべての言語に翻訳されているわけではありません。パブリック・キー証明書ログインを使用する際、時折、英語以外のバージョンの DCE に英語のメッセージが表示されることがあります。

DCE の各バージョン間のパブリック・キーの相互運用性

表19 は、1 つのセル内の異なる DCE バージョン間で、ログイン要求がどのように処理されるかを示しています。この表には、バージョン 1.1 またはそれ以降のサーバーのみが含まれています。これは、1.1 より前のサーバーはログイン要求の事前認証データを無視して、 DCE バージョン 1.1 より前の (事前認証なし) 応答を返すためです。

表 19. DCE 認証の相互運用

ログイン要求のタイプ	バージョン 1.1 および 1.2 のサーバーの応答	バージョン 1.2.2a のサーバーの応答
DCE バージョン 1.0		
任意のクライアントから。	事前認証。 *pre_auth_req* ERA インスタンスをチェックします。 ERA が存在しない場合、または既存の ERA が *value*= **0** (**NONE**) である場合、 DCE バージョン 1.0 (事前認証なし) 応答を返します。そうでなければ、ログイン要求を拒否します。	事前認証。 *pre_auth_req* ERA インスタンスをチェックします。 ERA が存在しない場合、または既存の ERA が *value*= **0** (**NONE**) である場合、 DCE バージョン 1.0 (事前認証なし) 応答を返します。そうでなければ、ログイン要求を拒否します。
TIMESTAMPS		

表 19. DCE 認証の相互運用 (続き)

ログイン要求のタイプ	バージョン 1.1 および 1.2 のサーバーの応答	バージョン 1.2.2a のサーバーの応答
DCE バージョン 1.1 およびこれ以上のクライアントから。	事前認証。 *pre_auth_req* ERA インスタンスをチェックします。 ERA が存在しない場合、あるいは既存の ERA が *value*= **0** (**NONE**) または *value*=**1** (**PADATA-ENC-TIMESTAMPS**) である場合、 DCE バージョン 1.1 の **TIMESTAMPS** 応答を返します。既存の ERA が *value*=**2** (**PADATA-ENC-THIRD-PARTY**) をもつ場合、ログイン要求を拒否します。 1.2 サーバーの応答: ERA が *value*=**3** **(PADATA-ENC-PUBLIC-KEY)** をもつ場合にも、ログイン要求を拒否します。	事前認証。 ERA インスタンスをチェックします。 ERA が存在しない場合、あるいは既存の ERA が *value*=**0 (NONE)** または *value*=**1 (PADATA-ENC-TIMESTAMPS)** である場合、 DCE バージョン TIMESTAMPS 応答を返します。既存の ERA が *value*=**2 (PADATA-ENC-THIRD-PARTY)**、 *value*=**3 (PADATA-ENC-PUBLIC-KEY)**、または *value*=**4 (PADATA-ENC-PUBLIC-KEY-CERTIFICATE)** の場合は、ログイン要求を拒否します。
THIRD -PARTY		
DCE バージョン 1.1 およびこれ以上のクライアントから。	1.1 サーバーの応答: 事前認証。 DCE バージョン 1.1 の **THIRD-PARTY** 応答を返します。 1.2 サーバーの応答: 事前認証。 *pre_auth_req* ERA インスタンスをチェックします。 ERA が存在して、 *value*=**3** **(PADATA-ENC-PUBLIC-KEY)** であれば、ログイン要求を拒否します。そうでなければ、 **THIRD-PARTY** 応答を返します。	事前認証。 *pre_auth_req* ERA インスタンスをチェックします。 ERA が存在して、 *value*=**3 (PADATA-ENC-PUBLIC KEY)** または *value*=**4 (PADATA-ENC-PUBLIC KEY -CERTIFICATE)** であれば、ログイン要求を拒否します。そうでなければ、 **THIRD-PARTY** 応答を返します。
PUBLIC- KEY		
DCE バージョン 1.2.2 クライアントから (ただし、 IBM DCE for Windows NT® バージョン 2.2 クライアントを除く)、および DCE for AIX バージョン 2.3 クライアント DCE レベル 2.2.0.4 以上 (実際の APAR 番号については、サービス技術員に問い合わせてください)。	1.1 サーバーの応答: 事前認証。 DCE バージョン 1.1 の **THIRD-PARTY** 応答を返します。 1.2 サーバーの応答: 事前認証。 DCE バージョン 1.2.2 の **PUBLIC-KEY** 応答を返します。	事前認証。 *pre_auth_req* ERA インスタンスをチェックします。 ERA が存在して、 *value*=**4 (PADATA-ENC-PUBLIC-KEY-CERTIFICATE)** であれば、ログイン要求を拒否します。そうでなければ、 DCE バージョン 1.2.2 の **PUBLIC-KEY** 応答を返します。
PUBLIC-KEY-CERTIFICATE		

表 19. DCE 認証の相互運用 (続き)

ログイン要求のタイプ	バージョン 1.1 および 1.2 のサーバーの応答	バージョン 1.2.2a のサーバーの応答
IBM DCE for Windows NT バージョン 2.2 クライアントから、あるいは DCE for Solaris, Version 2.3 クライアント または DCE for AIX バージョン 2.3 または 2.2 クライアント DCE レベル 2.2.0.4 以上から (実際の APAR 番号については、サービス技術員に問い合わせてください)。	1.1 サーバーの応答: 事前認証。 DCE バージョン 1.1 の **THIRD-PARTY** 応答を返します。 1.2 サーバーの応答: 事前認証。 ERA *pre_auth_req* インスタンスをチェックします。 ERA が存在して、 *value*=**3 (PADATA-ENC-PUBLIC-KEY)** であれば、ログイン要求を拒否します。そうでなければ、**THIRD-PARTY** 応答を返します。	事前認証。 **PUBLIC-KEY-CERTIFICATE** 応答を返します。

無効なログインの管理

プリンシパルに **2** (**PADATA-ENC-THIRD-PARTY**) の事前認証レベルを指定すると、セキュリティー・サーバーは、そのプリンシパルに対する無効なログインの試みを検出し、追跡することができます。これにより、管理者は、以下のことを行うことによって、パスワードを推測する攻撃の影響を制限できます。

- プリンシパルのアカウントが使用不能になるまでの、無効なログインの連続試行の回数に限度を設定する。 (ログインが成功すると、カウンターはリセットされます。)
- その限度に達した後、プリンシパルのアカウントを使用不能にする期間を指定する。

これは、プリンシパルに 2 つの ERA のインスタンス (*max_invalid_attempts* と *disable_time_interval*) を付加することによって行います。これらの ERA には次の値を指定します。

max_invalid_attempts
: プリンシパルのアカウントを使用不能とマークするまでにセキュリティー・サーバーが受け入れる、連続した無効なログイン試行回数を示す整数を指定します。

disable_time_interval
: プリンシパルのアカウントのログインの試行を禁止する秒数を指示する、整数を指定します。

次の例は、**dcecp** コマンドを使用してプリンシパルを作成し、 *max_invalid_attempts* および *disable_time_interval* ERA を付加する例です。

```
dcecp> principal create smitty -attribute {{max_invalid_attempts
7} ¥
> {disable_time_interval 60}}
dcecp>
```

注: DCE バージョン 1.1 の無効ログイン処理機能は、マスターが 1 つでレプリカがないセルにはログインの活動を正確に記録しますが、複製されたセルには正確なカウントを保持しません。これは、次の理由によります。

- 複製されたセルでは、ログイン試行はマスターまたはレプリカのいずれかにランダムに割り当てられます。
- 現在のところ、レプリカがマスターと通信するメカニズムはありません。したがって、マスターには正確なカウントを保持する手段がありません。

dcecp を使用してプリンシパルに ERA を付加する方法の詳細については、 417ページの『第32章 拡張レジストリー属性の作成および使用』を参照してください。

パスワードの強さとパスワード生成の管理

447ページの『第35章 ポリシーと属性の保守』に説明されている DCE パスワード・フォーマット・ポリシーによって、ユーザー・パスワードの次の特性を制御できます。

- 最小パスワード長
- パスワードをすべてスペースとすることができるかどうか
- パスワードを英数字だけで構成できるかどうか

カスタマイズされたパスワード検査および生成を行うパスワード管理サーバーを作成することによって、セル内のパスワードの強さのポリシーを拡張できます。 DCE はサンプルのパスワード検証 / 生成サーバー **pwdstrn2** を提供しており、ユーザーのセル要件に合わせたパスワード管理サーバーのベースとしてこれを使用できます。また、 DCE はパスワード強化サーバー **pwd_strengthd** を提供します。

DCE はパスワード管理 API を提供しており、アプリケーション開発者はこの API を使用して、プリンシパルのパスワード管理ポリシーについての情報を入手し、生成されたパスワードをパスワード管理サーバーに対して要求できます。パスワード管理 API の詳細については、*IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* を参照してください。

このサーバーを作成すれば、プリンシパルのパスワードが作成および変更されたときには、常にこのサーバーの妥当性検査を受けるようにすることができます。次に示すように、*pwd_val_type* および *pwd_mgmt_binding* ERA のインスタンスをプリンシパルに付加することによってこれを行います。

pwd_val_type
プリンシパル用のパスワード作成オプションを次のように指定します。

0 **(NONE)**。プリンシパルのパスワードが DCE 標準ポリシーにのみ従うことを指定します。 (DCE 標準ポリシーの説明については、 447ページの『第35章 ポリシーと属性の保守』を参照してください。) **0** (**NONE**) を指定することは、プリンシパルに ERA インスタンスを付加しないことと同じです。

1 **(USER_SELECT)**。プリンシパルが、 *pwd_mgmt_binding* ERA に指定されているパスワード管理サーバーへの入力として、パスワード・テキストを与えなければならないことを指定します。

2 **(USER_CAN_SELECT)**。プリンシパルがパスワードを提供する

か、または *pwd_mgmt_binding* ERA で指定されたパスワード管理サーバーの生成するパスワードを選択するかのいずれかであることを指定します。

3 **(GENERATION_REQUIRED)**。 *pwd_mgmt_binding* ERA に指定されているパスワード管理サーバーがプリンシパルのパスワードを生成すべきことを指定します。

pwd_mgmt_binding

セルのパスワード管理サーバーへのバインディングを指定します。 *pwd_mgmt_binding* ERA を構成する項目は次のとおりです。

認証サービス

認証サービスのタイプを指定します。常に **dce** を使用してください。

パスワード強化サーバー・プリンシパル

パスワード強化サーバー・プリンシパルの名前を指定します。 **-server_princ** オプションで指定した名前と同じ名前を使用してください。デフォルトは **pwd_strengthd** です。

保護レベル

このバインディングを使用して行われたリモート・プロシージャー・コールの保護レベルを指定します。保護レベルには、 **pktprivacy**、**cdmf**、および **pktinteg** があります。

認証レベル

認証レベルを指定します。常に **secret** を使用してください。

認可サービス

サーバーよって実装される認可サービスを指定します。常に **name** を使用してください。

cds オブジェクト名

バインディングを指定します。

次に示す例は、**dcecp** コマンドを使用してプリンシパルを作成し、 *pwd_val_type* および *pwd_mgmt_binding* ERA を付加する例です。

```
dcecp> principal create smitty -attribute {{pwd_val_type 2} ¥
> {pwd_mgmt_binding ¥
> {{dce pwd_strengthd pktprivacy secret name} ¥
> {/.:/subsys/dce/pwd_mgmt/pwd_strengthd}}}
dcecp>
```

dcecp を使用してプリンシパルに ERA を付加する方法の詳細については、 417ページの『第32章 拡張レジストリー属性の作成および使用』を参照してください。パスワード変更時の、生成されたパスワード要求に関する情報は、 380ページの『dcecp によるパスワードの生成』を参照してください。

パスワード管理サーバーの構成については、以下のセクションおよび *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド － 入門* を参照してください。

パスワード管理サーバーの管理

IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド ― 入門 の第 2 部で、**config.dce** を使用してパスワード管理サーバーを構成する方法について説明しています。このセクションでは、パスワード管理サーバーの管理についてのその他の注意を述べます。

- パスワード・セキュリティーを保護し、またパフォーマンスを最適化するには、パスワード管理サーバーをマスター DCE セキュリティー・サーバーと同じマシンで実行する必要があります。
- パスワード管理サーバーのデフォルトのパス名は、 **opt/dcelocal/bin** です。
- **config.dce** は、セル内にパスワード管理サーバーをただ 1 つ構成することをサポートします。追加のサーバーは手動で構成できます。その後、各プリンシパルの適切なサーバーを指すように、プリンシパル *pwd_mgmt_binding* ERA をセットすることができます。
- パスワード管理サーバーを別のバージョンに置き換えるには、以下の手順に従ってください。
  1. **pwd_strengthd** を停止します。
  2. **opt/dcelocal/bin/pwd_strengthd** を名称変更します。
  3. 新しいサーバーを、 **opt/dcelocal/bin/pwd_strengthd** にコピーします。
  4. **pwd_strengthd** を開始します。
- パスワード管理サーバーのログ・ファイルは、**opt/dcelocal/var/security/pwd_strength** ディレクトリーにあります。ログ・ファイルの名前は、 **-server_princ** オプションで指定した名前から生成されます。**-server_princ** オプションを指定しない場合は、デフォルトのログ・ファイル名 **pwd_strengthd.log** になります。

dcecp によるパスワードの生成

プリンシパルに、値 **2** (**USER_CAN_SELECT**) または **3** (**GENERATION_REQUIRED**) をもつ *pwd_val_type* ERA が存在するとき、そのプリンシパルは、パスワード変更時にパスワード生成を要求することができます (または要求されます)。プリンシパル **smitty** の場合、これを行うには、次のような一連の **dcecp** コマンドが使用できます。

```
dcecp> set p [account generate smitty]
newgenpwd
dcecp> account modify smitty -password $p -mypwd -dce-
dcecp>
```

このコマンドは、パスワード管理サーバーに対して生成されたパスワードを要求し、新規パスワードを *p* 変数に入れ、それを画面へ出力します (*newgenpwd*)。 (新規パスワードを確実に覚えてください。) 次に、*p* に保管した値を、 **account modify** または **account create** コマンドの新しいパスワードの値として渡します。

警告: 次の **dcecp** コマンドは絶対に実行しないでください。その理由は、パスワードがアカウント内で変更されますが、ユーザーには新しく生成されたパスワードが表示されないためです。

```
dcecp> account modify smitty -password [account gen smitty] -mypwd -dce-
dcecp>
```

パスワード強化サーバーを使用したパスワード変更

ユーザーは、**dce_login** コマンドの **dcecp**、 **rgy_edit**、または **-n** オプションを使用して、パスワードを変更することができます。セル管理者がユーザーのパスワードをリセットする必要がある場合は、 **rgy_edit** または **dcecp** コマンドを使用できます。

dcecp では、 *pwd_val_type* のタイプが **3** (ユーザーのパスワードをパスワード強化サーバーで生成することが必要) のユーザーは、上記 380ページの『dcecp によるパスワードの生成』 で定義した 2 つのステップのプロセスだけでパスワードを変更できます。

dce_login または **rgy_edit** コマンドは、新規パスワードの入力を促す前に、*pwd_val_type* が **2** または **3** のユーザー用に生成されているパスワードを表示します。

IBM DCE 拡張パスワード強化サーバー

IBM DCE 拡張パスワード強化サーバーは、拡張されたパスワード検査を実行します。このサーバーは、合成規則、経過時間規則、再使用 (ヒストリー) 規則、辞書リスト、およびユーザー定義の規則に従って、パスワード妥当性検査を行うことができます。拡張パスワード強化サーバーは *IBM_pwd** ERA を使用して、実施する規則を選択します。この ERA は、ユーザー固有レベル、オーガニゼーション・レベル、またはサーバー・レベルでセットできます。

アカウントが拡張されたパスワード検査をできるようになる前に、 *pwd_val_type* および *pwd_mgmt_binding* ERA を使用して、プリンシパルをパスワード検査サーバーに定義しなければなりません。 **dcecp** コマンドの例については、 379ページを参照してください。プリンシパルがパスワード強度検査またはパスワード生成 (あるいはその両方) を使用できるようになると、パスワード検査サーバーは *IBM_pwd** ERA を使用して、実施する規則を選択することができます。

注: DCE の旧バージョンのパスワード強化サーバーで使用できたいくつかのコマンド行オプションは廃止されました。廃止されたオプションは、次のとおりです。

- **+/-all_spaces**
- **+/-alpha_num**
- **-min_len**

これらのオプションは互換性を保つために拡張サーバーでも機能しますが、これらのオプションは使用しないでください。代わりに、拡張サーバーはレジストリーから類似のパスワード規則を読み取り、その規則を使用してユーザー・パスワードを検査できます。これは、dcecp コマンド **registry modify** および **organization modify** を使用して、全レジストリーに適用されるパスワードまたはオーガニゼーション固有のパスワードを設定することによって行えます。

ERA の規則

各グループごとの ERA カテゴリーおよび属性は、次のように定義されます。

IBM_pwd_comp_rules ERA: この合成検査 ERA は、パスワードに使用できる文字のタイプや組み合わせを指定します。以下の規則をサポートすることができます。

mindiff
新規パスワード内の文字のうち、旧パスワード内の文字と異なる必要のある文字数の最小値。

maxrepeat
パスワード内で特定の文字を繰り返し使用できる回数の最大値。

minalpha
パスワード内に使用すべき英字の文字数の最小値。

minother
パスワード内に使用すべき英字以外の文字数の最小値。

minlen
パスワード内に使用すべき文字数の最小値。

maxlen
パスワード内に使用できる文字数の最大値。

それぞれの規則は、一連の attribute=value の組を使って定義します。

たとえば、次のようにします。

```
dcecp> principal modify smitty -add {IBM_pwd_comp_rules mindiff=4 maxrepeat=2 ¥
minalpha=4 minother=3 minlen=7 maxlen=8}
```

注: IBM パスワード強化サーバーは、以下の場合に **maxlen** をリセットします。

- **minalpha** と **minother** の合計が **maxlen** より大きい場合、 **maxlen** は **minalpha** + **minother** に設定されます。
- **minlen** が **maxlen** より大きい場合、 **maxlen** は **minlen** と等しくなるように設定されます。

識別できない属性名が ERA にあれば、それらは無視されます。

IBM_pwd_age_rules ERA: この経過時間検査 ERA によって、パスワードが変更できるようになるまでの期間を指定できます。以下の規則をサポートすることができます。

minage
パスワードを変更できるようになるまでに、そのパスワードを使用しなければならない最小時間。最小経過時間の規則は、パスワードを変更できるようになるまでに、そのパスワードを使用しなければならない時間の最小値を定義します。アカウントのパスワードが最後に変更された時刻のタイム・スタンプは、既にセキュリティー・レジストリーに保管されています。この情報を入手し、現在時刻と比較することによって、ユーザーのパスワード変更を許可するかどうかが決定されます。時間は週数で指定します。

それぞれの規則は、一連の attribute=value の組を使って定義します。

たとえば、次のようにします。

```
dcecp> principal modify smitty -add {IBM_pwd_age_rules minage=1}
```

識別できない属性名が ERA にあれば、それらは無視されます。

IBM_pwd_hist_rules ERA: ヒストリー検査 - あるパスワードを再び使用できるようになるまでの時間、および、あるパスワードを再使用できるようになるまでに使わなければならない、異なるパスワードの数。ユーザーの以前のパスワードは、所定の基準を満たすまでは、新規パスワードとして再使用できません。以下の規則をサポートすることができます。

histexpire
以前のパスワードが再使用できるようになるまでに経過しなければならない時間の定義。指定する時間は、週の数です。

histsize
以前のパスワードを再使用できるようになるまでに必要な、パスワード変更の所定の回数。

注: ヒストリー・データベースのスペース節約のために、ヒストリー規則を実施する必要のあるパスワードだけが、ここに保管されます。

注: **mindiff**、**minage**、**histexpire**、および **histsize** については、新規パスワードを現行パスワードと比較するための規則では、規則を適用する前にパスワード強化サーバーでパスワードが 1 回変更されている必要があります。検査ルーチンが、パスワード強化ヒストリー・データベースから現行パスワードを検索します。現行パスワードがパスワード強化サーバーを使用して変更されていない場合、現行パスワードはヒストリー・データベースには存在しません。この場合、新規パスワードはブランクと比較されるので、この操作は失敗します。

それぞれの規則は、一連の attribute=value の組を使って定義します。

たとえば、次のようにします。

```
dcecp> principal modify smitty -add {IBM_pwd_hist_rules histexpire=26 histsize=25}
```

識別できない属性名が ERA にあれば、それらは無視されます。

IBM_pwd_dict_rules ERA: この辞書検査 ERA は、パスワード妥当性検査で使用する辞書ファイルのリストを指定します。辞書ファイルには、パスワードとして使用できない言葉のリストが収められています。たとえば、管理者は、 (社名など) 容易に推測できるいくつかのことばをユーザーに使わせたくない場合があります。さらに、管理者は個々のユーザーに対して、 (家族の名前など) 人を意味する言葉の使用を制限したい場合があります。この ERA の値は、パスワード妥当性検査に使用される辞書ファイルのリストです。

たとえば、次のようにします。

```
dcecp> principal modify smitty -add {IBM_pwd_dict_rules dictionary.english}
```

ファイル・フォーマット: 辞書ファイルのフォーマットは、以下のような AIX および Solaris Dictionary File フォーマットと同じです。

- 1 行につき 1 語
- 各語は第 1 列から始まり、改行文字で終わる
- 組み込み、先行、および後続の空白文字は、はく奪**されない**。

保管場所: 辞書ファイルは、このパスワード強化サーバーが稼働しているマシン上に保管しなければなりません。辞書ファイルの指定には、絶対パスまたは相対パスを使用できます。相対パスの場合、以下のように、サーバーのデータ・ディレクトリーからの相対位置を指定します。

```
/opt/dcelocal/var/security/pwd_strength
```

パスワード強化サーバーを新しいマシンに移動した場合、必ず辞書ファイルも転送してください。

欠落ファイル: パスワード強化サーバーが指定されたファイルを見付けられない場合、またはファイルにアクセスできない場合は、エラーがログに記録されて、パスワードを変更することはできません。

識別できない属性名や値が ERA にあれば、それらはエラーとしてログに記録されます。

IBM_pwd_userdef_rules ERA: このユーザー定義規則 ERA を使用すると、提供された規則だけでは必要が満たされない場合に、管理者は独自のパスワード検査ルーチンを作成できます。たとえば、各パスワードの 2 番目の文字が数字でなければならないという規則を設けたい場合があります。この ERA の値は、パスワード妥当性検査に使用される C 共用オブジェクトでなければなりません。

注: ユーザー定義の検査を実行するには、 **-userdef** オプションを使用して拡張パスワード強化サーバーを開始しなければなりません。こうしない場合は、エラー・メッセージが生成されて、パスワードの変更は行えません。

たとえば、次のようにします。

```
dcecp> principal modify smitty -add {IBM_pwd_userdef_rules pw.checker}
```

ユーザー定義規則に従ったパスワードの検査は、指定された共用オブジェクトをランタイムでロードして、情報をエントリー・ポイントに渡すことによって行われます。エントリー・ポイントは、以下のように定義された関数でなければなりません。

```
int pwdrestrict_method(char *user, char *new_passwd, char *old_passwd, char **message);
```

パスワードを制限するすべての方法に適用される規則として、メッセージを呼び出し側のサブルーチンに返します。 stdout や stderr にはメッセージを出力しません。呼び出し側プログラムに対して動的に割り振られたメモリーにメッセージを返さなければならないことに注意してください。呼び出し側プログラムは、メモリーを使用し終わるとメモリーを割り振り解除します。

ロード可能なサブルーチン・モジュールに関しては、以下のようないくつかの注意事項があります。

1. *NewPassword* および *OldPassword* の値は、ユーザーが実際に入力した明確なテキスト・パスワードです。これらのパスワードをメモリーの他の部分にコピーする場合は、呼び出し側プログラムに戻る前に、それらのメモリー位置を消去してください。こうすると、明確なテキスト・パスワードがメモリー内のダンプに現れなくなります。また、別のプログラムがアクセスできるファイルおよびその他

の場所に、これらのパスワードをコピーしないでください。明確なテキスト・パスワードを、プロセス・スペース以外の場所に決して残さないようにしてください。

2. プロセスのシグナル・ハンドラーの現在の設定値を変更しないでください。
3. (たとえば **exit** サブルーチンや **exec** サブルーチンのような) プログラムの実行を終了させる関数を呼び出さないでください。常に呼び出し側プログラムに戻るようにしてください。
4. コードはスレッド・セーフでなければなりません。
5. 実際のロード・モジュールは、書き込み禁止の環境で保管する必要があります。ロード・モジュールおよびディレクトリーは、 root ユーザーのみが書き込み可能にしてください。

ユーザー定義の検査方法は、サーバーが最後に実施する制限です。

パラメーター:

UserName
ユーザー名を指定します。

NewPassword
新規パスワードを明確なテキストで指定します。この値は NULL ポインターの場合があります。

OldPassword
現在のパスワードを明確なテキストで指定します。この値は NULL ポインターの場合があります。

Message
エラー・メッセージを含む **malloc** されたメモリーへのポインターのアドレスを指定します。 **malloc** されたメモリーとメッセージは、この検査方法によって与えられることになっています。

リターン値: この方法は、以下のような値を返します。リターン値は、優先順位の順番でリストされています。

-1 内部エラー。この方法によってパスワード評価を実行できませんでした。(メッセージ用のメモリーが割り振り不能でない限り) この方法によって、エラー・メッセージが *Message* に書き込まれます。メモリーを割り振ることができない場合は、 NULL ポインターを *Message* に返します。

1 失敗。パスワード変更は、制限のための要件を満たしませんでした。パスワード制限が正しく評価されて、パスワード変更は受け入れられませんでした。この方法によって、エラー・メッセージが *Message* に書き込まれます。

0 成功。パスワード変更は、制限のための要件を満たしました。

以下の注記は、AIX 固有の情報です。

注:

以下の入力は、コンパイラーおよびローダーに対して、入力ファイルをコンパイルして、その結果できたオブジェクト・ファイルのエントリー・ルーチンとして、指定された検査ルーチンを使用するよう指示します。

cc -e *checking_routine* **-o** *out_file input_file*

たとえば、次のようにします。

cc -e *pwdrestrict_method* **-o** *pwd_check.libpwd_checker.c*

この関数についての詳しい情報は、 AIX Version 4.3 Base Operating Systems and Extensions Technical Reference 第 1 巻の **pwdrestrict_method サブルーチン**のセクションを参照してください。このインターフェースを使用すると、すでに作成した User-Defined Rules for AIX Security を変更せずに、引き続きこれらの規則を使用できます。

以下の注は、Solaris だけのものです。

注:

以下の入力は、C コンパイラーおよびローダーに対して、**pwdrestrict_method ルーチン**が入っている入力ファイルをコンパイルして、その結果できたファイルのロード可能ライブラリーを作成するよう指示します。

cc -G -o *out_fileinput_file*

たとえば、次のようにします。

cc -G -o *pwd_check.lib pwd_checker.c*

規則の定義: ユーザー定義規則は、辞書規則と同様にして実装されます。 ERA (**IBM_pwd_userdef_rules**) を使用して、パスワード検査のこの段階で使う共用オブジェクトのファイル名を保持します。

保管場所: 共用オブジェクト・ファイルは、このパスワード強化サーバーが稼働しているマシン上に保管しなければなりません。共用オブジェクト・ファイルの指定には、絶対パスまたは相対パスを使用できます。相対パスの場合、サーバーのデータ・ディレクトリー **/opt/dcelocal/var/security/pwd_strength** からの相対位置を指定します。

パスワード強化サーバーを新しいマシンに移動した場合、必ず共用オブジェクト・ファイルも転送してください。

欠落ファイル: パスワード強化サーバーが指定された共用オブジェクトを見付けられない場合、またはファイルにアクセスできない場合は、エラーがログに記録されて、パスワードを変更することはできません。

識別できない属性名や値が ERA にあれば、それらはエラーとしてログに記録されます。

ユーザー固有、オーガニゼーション固有、およびサーバー固有の値

ユーザー固有の値については、 ERA がユーザーのプリンシパルに直接付加されます。オーガニゼーション固有の ERA については、 ERA をオーガニゼーション ID に付加する必要があります。サーバー固有の ERA については、 ERA をパスワード強化のプリンシパルに付加する必要があります。これは、**-server_princ** オプションで指定したパスワード強化サーバーの名前です。サーバー固有およびオーガニゼーション固有の規則は、パスワード強化サーバーに定義されたすべてのユーザー

のいずれのパスワード変更に対しても、検査が実施されます。 1 つのユーザーに対して複数の規則を定義した場合の競合の解決方法については、 『規則のマージ』を参照してください。

サーバー固有の規則の例:

```
dcecp> principal modify pwd_strengthd -add {IBM_pwd_comp_rules minlen=8 maxlen=10}
```

注: パスワード・サーバー・プリンシパルは、 **-server_princ** オプションを使って構成できます。

オーガニゼーション固有の規則の例:

```
dcecp> organization modify area2 -add {IBM_pwd_dict_rules dictionary.english}
```

パスワードの生成

パスワード生成ルーチンが変更されて、生成されたパスワードの妥当性検査を実行するようになりました。生成されたパスワードのうち、ユーザー、オーガニゼーション、およびサーバー固有の規則による妥当性検査に合格したものだけが、パスワード・キャッシュに保管されます。

pwd_mgmt_binding ERA (つまり、 subsys/dce/pwd_mgmt/pwd_strengthed) で */.:* セル名省略を使用する場合は、パスワード生成はセル間境界を超えて機能することはありません。この制限があるために、 **pwd_val_typd** ERA が 2 または 3 であるプリンシパルのパスワードをセル間接続を介して変更することはできません。

この問題を避けるために、正規の完全セル名を使用してバインディングを指定してください。

たとえば、次のように指定できます。

```
dcecp -c principal modify princ_name -add {pwd_mgmt_binding  {{ dce pwd_strengthd ¥
pktprivacy secret name}/.../cell_name/subsys/dce/pwd_mgmt/pwd_strengthd}}
```

このコマンドは、パスワードの変更が実行されたときに、 *princ_name* をデフォルトのローカル・セルではなく、 *cell_name* のパスワード強化サーバーに明示的に送信します。

規則のマージ

IBM DCE 拡張パスワード強化サーバーでは、ユーザー、オーガニゼーション、サーバー、またはそれらすべてに対して規則が定義されている場合、最も厳格な値を組み合わせた規則が実施されます。最も厳格な規則は、以下の 表20 に従って実施されます。

表 20. 規則のマージ

規則	最も厳格な値
mindiff	最大値
maxrepeats	最小値
minalpha	最大値
minother	最大値
minlen	最大値
maxlen	最小値

表 20. 規則のマージ (続き)

規則	最も厳格な値
minage	最大値
histexpire	最大値
histsize	最大値

注: 辞書リストおよびユーザー定義リストは連結されます。ユーザー、オーガニゼーション、およびサーバーに対して定義されている場合、すべてのリストがパスワード妥当性検査で使われます。

パスワードの有効期限の管理

デフォルトでは、DCE セキュリティー・サーバーはパスワードの有効期限が切れたプリンシパルのログインを不能にします。しかし、このようにしたくない場合もあります。たとえば、パスワードの有効期限が切れたために **cell_admin** がセルからロックアウトされることを避けたい場合があります。 secd パスワードの有効期限が切れた場合、または自分のプリンシパルのパスワードの有効期限が切れた場合、secd および dced は認証することができず、デーモンは操作できません。

推奨されるモデルは、これらの制限を「永久」にセットし、オーガニゼーションまたはアカウントにしたがって細分化することです。 secd パスワードの有効期限が切れた場合、または自分のプリンシパルのパスワードの有効期限が切れた場合は、secd locksmith モードに入り、必要な変更を行います。

特定のプリンシパルのパスワード有効期限検査を管理するには、プリンシパルに *passwd_override* ERA のインスタンスを付加し、次の値のいずれか 1 つを指定します。

0 (**NONE**)。プリンシパルのパスワード有効期限検査をオーバーライドしないことを指定します (つまり、プリンシパルが有効期限の切れたパスワードを使ってログインすることを禁止します。) **0** (**NONE**) を指定することは、プリンシパルに ERA インスタンスを付加しないことと同じです。

1 (**OVERRIDE**)。プリンシパルのパスワード有効期限検査をオーバーライドすべきことを指定します (つまり、プリンシパルが有効期限の切れたパスワードを使ってログインすることを許可します。)

次の例は、プリンシパルを作成し、 *passwd_override* ERA を付加する、 **dcecp** コマンドの例です。

```
dcecp> principal
create smitty -attribute {passwd_override 1}
dcecp>
```

dcecp を使用してプリンシパルに ERA を付加する方法の詳細は、 417ページの『第32章 拡張レジストリー属性の作成および使用』を参照してください。

グループおよびオーガニゼーションの追加と保守

グループまたはオーガニゼーションをアカウントで使用するには、その前にグループまたはオーガニゼーションをレジストリーに追加しなければなりません。 **dcecp group create** を使用してグループを追加する際、個々のグループをプロジェクト・リストに含めるかどうかを制御する、プロジェクト・リスト包含属性をセットできます。 (プロジェクト・リストは、オーガニゼーションには適用されません。)

プロジェクト・リスト

プリンシパルのプロジェクト・リストとは、そのプリンシパルまたは別名がメンバーとなっている、すべてのグループのリストです。プリンシパルがオブジェクトにアクセスを試行する時、プリンシパルはオブジェクトの ACL で指定されているすべてのグループのメンバーシップからアクセス権を与えられます。 (ACL の説明は、329ページの『第28章 アクセス制御リストの使用』を参照してください。) たとえば、ファイル X の ACL にエントリーが 2 つ含まれていると仮定します。 1 つはグループ A に書き込みアクセスを許可し、もう 1 つは、グループ B に読み取りアクセスを許可します。この場合、グループ A と B の両方のメンバーであるプリンシパルは誰でも、ファイル X の読み取りと書き込みを行うことができます。

プロジェクト・リストおよび権利

プリンシパルは、ログイン時に使用した名前に関連したグループからのみプロジェクト・リストのアクセス権を取得します。プリンシパルの名前、およびそれらの別名からは権利を取得しません。たとえば、**gustav** という名前のプリンシパルがグループ A と B のメンバーであるとします。別名 **gus** のもとでは、 **gustav** はグループ C と D のメンバーでもあります。プリンシパルは、**gustav** としてログインすると、グループ A と B からのみアクセス権を与えられます。プリンシパルが、別名 **gus** としてログインすると、グループ C と D からのみアクセス権を与えられます。

プリンシパル (またはその別名) がメンバーとなっているグループを表示するには、439ページの『第34章 レジストリー情報の表示』に説明されている、 **principal show** コマンドを使用します。

プロジェクト・リストへの包含の禁止

グループがプロジェクト・リストに含めることが禁止されている場合、その権利は取得できません。たとえば、ファイル X の ACL に 2 つのエントリーが含まれていると仮定します。 1 つは、グループ A にファイル X への読み取りアクセスを許可し、もう 1 つは、グループ B にファイル X への書き込みアクセスを許可しています。プロジェクト・リスト包含属性によって、グループ B をプロジェクト・リストに含めることを禁止するようにセットされていると仮定します。グループ A と B の両方のメンバーであり、ファイル X にアクセスしようとするプリンシパルは、読み取りだけは許可されますが、書き込みは許可されません。プロジェクト・リスト包含属性が、グループ B をプロジェクト・リストに含めることを許可している場合、グループ A と B のメンバーは、読み取りと書き込みの両方のアクセス権を受け取ります。

あるグループをリストに含めないようにすることができます。たとえば、ルートと同じアクセス権を持つ予約されたグループがプロジェクト・リストに含まれないようにすることができます。

グループおよびオーガニゼーションの追加

グループを追加するには、**dcecp group create** コマンドを使用し、オーガニゼーションを追加するには、 **dcecp organization create** コマンドを使用します。グループまたはオーガニゼーションを追加するときは、グループまたはオーガニゼーションの 1 次名を指定する必要があります。さらに、表21 にリストされている属性オプションを指定することができます。

dcecp group create コマンドおよび **dcecp organization create** コマンドを使用するとき、複数のプリンシパルを作成するのと同じように、 1 つのコマンドで複数のグループまたは複数のオーガニゼーションを作成することができることに注意してください。詳細については、 361ページの『プリンシパルの追加』を参照してください。

表 21. グループおよびオーガニゼーションを作成するための属性オプション

情報	意味
-gid	グループまたはオーガニゼーションに関連した必須の UNIX ID。この番号は、ユーザーが明示的に入力するか、または自動的に生成することができます。入力する数字は、**dcecp registry modify** コマンドでセットされた、最大許容 UNIX 番号 (**maxuid** 属性) を超えることはできません。ただし、**registry modify** コマンドでグループまたはオーガニゼーションにセットされた下限の UNIX 番号 (**minuid** 属性) より小さな数字は入力できます。番号が自動的に割り当てられるようにすると、番号は、下限 UNIX 番号と最大 UNIX 番号で定義された範囲内に入ります。
-fullname *string*	1 次名をさらに完全に記述するために使用するオプショナル名。スペースを含めるには、フルネームを中括弧で囲みます。デフォルトはブランクです。
-inprojlist *value*	グループに対してのみ適用され、グループをプロジェクト・リストに入れるかどうかを示します。デフォルトは、**yes** です。

グループの追加

次の例は、**symphonists** という名前のグループを **レジストリー**に追加する方法を示しています。

```
dcecp> group create symphonists
dcecp>
```

この例では、グループ UNIX ID は自動的に生成され、フルネームは与えられず、グループはプロジェクト・リストに含まれます。

オーガニゼーションの追加

次の例は、**classic** という名前のオーガニゼーションをレジストリーに追加する方法を示しています。

```
dcecp> organization create classic
dcecp>
```

この例では、オーガニゼーションの UNIX ID は自動的に生成され、フルネームは与えられません。

グループおよびオーガニゼーションの変更

グループとオーガニゼーションの場合、1 次名とフルネームを変更することができます。さらに、グループについては、プロジェクト・リストにグループを入れるどうかを変更でき、オーガニゼーションについては、ポリシーを変更できます。 (オーガニゼーション・ポリシーの変更の詳細については、 447ページの『第35章 ポリシーと属性の保守』を参照してください。)

グループの変更には、**dcecp group modify** コマンドを使用します。次の例は、このコマンドを **-inprojlist** のオプションとともに使用して、グループ **symphonist** のプロジェクト包含属性を **yes** (プロジェクト・リストに含める) から **no** (プロジェクト・リストに含めない) に変更するものです。

```
dcecp> group modify symphonists -inprojlist
no
dcecp>
```

グループの 1 次名を変更するには、**dcecp group rename** コマンドを使用し、オーガニゼーションの 1 次名を変更するには、 **dcecp organization rename** コマンドを使用します。これらのコマンドの形式は次のとおりです。

```
group
rename old_name -to new_name
organization rename old_name -to new_name
```

ここで、

old_name
変更するグループまたはオーガニゼーションの 1 次名。

new_name
グループまたはオーガニゼーションの新しい 1 次名。

次は、**group rename** コマンドを使用して 1 次名 **symphonists** を **symphonists7** に変更する例です。

```
dcecp> group rename symphonists -to symphonists7
dcecp>
```

1 次名を変更すると、その変更は、グループまたはオーガニゼーションがメンバーとなっている、すべてのグループとオーガニゼーションのメンバーシップ・リストにも反映されることに注意してください。

注: このリリースの DCE では、アグリゲートのプリンシパル、グループ、またはオーガニゼーション (pgo) を移動してそれ自身の下位にするような操作はサポートされません。たとえば、以下のような一連のレジストリー操作は使用できません。

```
dcecp -c group create foo/bar
dcecp -c group create foo
dcecp -c group rename foo -to foo/xyz
```

グループおよびオーガニゼーションの削除

グループあるいはオーガニゼーションを削除すると、そのグループあるいはオーガニゼーションを使用するアカウントも自動的に削除されます。たとえば、グループ **symphonists** を削除すると、アカウント **vivaldi symphonists baroque** および **mozart symphonists classic** も自動的に削除されます。

グループを削除するには **dcecp group delete** コマンドを使用し、またオーガニゼーションを削除するには **dcecp organization delete** コマンドを使用します。次の例は、**group delete** コマンドを使用して、グループ **symphonists** を削除する例です。

```
dcecp> group delete symphonists
dcecp>
```

次の例は、**organization delete** コマンドを使用して、オーガニゼーション **classic** を削除する例です。

```
dcecp> organization
delete classic
dcecp>
```

1 つの **group delete** または **organization delete** コマンドを使用して複数のグループまたはオーガニゼーションを削除するには、複数のプリンシパルを削除するのと同様に、削除する名前を中括弧で囲み、スペースで区切ります。

メンバーシップ・リストの保守

それぞれのグループやオーガニゼーションには、そのグループやオーガニゼーションのメンバーとなっているプリンシパルをリストした、メンバーシップ・リストがあります。グループのメンバーは、ローカル・セルまたは外部セルのどちらのプリンシパルでも差し支えありません。オーガニゼーションのメンバーは、ローカル・セルからのみのプリンシパルでなければなりません。メンバーシップ・リストにメンバーを追加するには、 **dcecp group add** コマンドを使用し、リストからメンバーを削除するには **dcecp group remove** コマンドを使用します。

グループまたはオーガニゼーションからメンバーを削除すると、削除されたメンバーのそのグループまたはオーガニゼーションに関連したアカウントも削除されます。たとえば、プリンシパル **mahler** をグループ **symphonists** から削除すると、アカウント **mahler symphonists classic** も削除されます。

グループあるいはオーガニゼーションからプリンシパルを削除すると、オブジェクトに対するプリンシパルの権利が影響を受ける場合があることに注意してください。この変更は、プリンシパルのチケット発行チケットが更新されたときに、初めて有効になります。チケット更新の詳細については、 395ページの『第31章 アカウントの作成および保守』を参照してください。

アカウント作成のメンバーシップ・リストへの影響

アカウントを作成するとき、アカウントが作成されるプリンシパルは、そのアカウントに指名されているグループあるいはオーガニゼーションのメンバーでなければなりません。たとえば、アカウント **mahler symphonists classic** を作成する場合、プリンシパル **mahler** は、 **symphonists** グループと **classic** オーガニゼーションのメンバーでなければなりません。

グループ・メンバーの追加と削除

次の例では、**dcecp group add** コマンドを **-member** オプションとともに使用して、 **mahler** をグループ **symphonists** に追加し、 **strauss** をグループ **symphonists** から削除します。

```
dcecp> group add symphonists -member mahler
dcecp> group remove symphonists -member strauss
dcecp>
```

プリンシパルの完全修飾名を指定すれば、外部セルのメンバーをローカル・グループに追加できることに注意してください。ただし、オーガニゼーションのメンバーは、ローカル・セルのみのプリンシパルでなければなりません。

1 つの **group add** または **group remove** コマンドによって、複数のメンバーを追加あるいは削除することができます。そうするには、メンバー名を中括弧で囲み、間をスペースで区切ります。たとえば、プリンシパル **bach**、**britten**、および **mahler** を、グループ **symphonists** に追加するには、次のように入力することができます。

```
dcecp> group add symphonists -member
{bach britten mahler}
dcecp>
```

ホストが DCE セルにログインされている間にホストのグループ名を変更するというまれなケースでは、追加的なステップを実行して新しいグループ名情報でホストの証明書を更新しない限り、既存のホストの証明書は無効になります。

ホストの証明書は、DCE ホスト上でセキュリティー・クライアント機能を実行する **secval** プロセスによって管理されます。通常、ホストのスタート直後に、 **secval** プロセスはホストを DCE セルにログインさせ、ホストの証明書を入手して、それらをホストに格納します。グループ名情報を変更したあと、**secval** プロセスを非活動化してから、再度活動化して、これらの証明書を更新します。次の例は、リモート・ホスト **persephone** でのこれらの操作を示しています。

```
dcecp> secval deactivate /.:/hosts/persephone/config/secval
dcecp> secval activate /.:/hosts/persephone/config/secval
dcecp>
```

プリンシパルまたはグループの別名の作成および保守

プリンシパルとグループの別名を作成し、保守するには、 **dcecp principal create** コマンドを使用します。オーガニゼーションには、別名を与えることができません。

別名の作成

プリンシパルの別名を作成するには、次の形式の **dcecp principal create** コマンドを入力します。

principal create *name* **-uid** *unix_ID* **-alias yes**

ここで、

name　プリンシパルまたはグループの別名。

unix_ID
　別名を作成するプリンシパルに関連した UNIX ID。

-alias　*name* が別名であることを示す。

グループの別名を作成するには、次の形式の **dcecp group create** コマンドを入力します。

```
group create name -gid group_ID -alias
yes
```

ここで、

name プリンシパルまたはグループの別名。

group_ID
別名を作成するグループに関連した UNIX ID。

-alias *name* が別名であることを示す。

1 次名から別名への変更、および別名から 1 次名への変更

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

別名を 1 次名にあるいは 1 次名を別名に変更するには、プリンシパルには **dcecp principal modify** コマンドを、またグループには **dcecp group modify** コマンドを、それぞれ使用します。これらのコマンドの形式は次のとおりです。

```
principal modify name -alias {yes|no}
group modify name -alias {yes|no}
```

ここで、

name 別名に変更する 1 次名、あるいは 1 次名に変更する別名。

-alias **-alias yes** は、 *name* によって識別される 1 次名を別名に変更します。
-alias no は、 *name* によって識別される別名を 1 次名に変更します。

プリンシパルまたはグループは、1 次名を一時点で 1 つしか持つことができません。別名を 1 次名に変更する前に、まず 1 次名を別名に変更してください。

第31章 アカウントの作成および保守

プリンシパルはすべて、2 つの識別を持っています。 1 つはネットワーク識別で、ネットワークを通じてマシンへのアクセスを可能にします。もう 1 つはローカル識別で、ローカル・マシンのオブジェクトへのアクセスを可能にします。 2 つの識別は、タンデムの形で存在しますが、お互いに独立しています。プリンシパルのネットワーク識別は、ネットワーク・レジストリー内のアカウントによって定義されます。プリンシパルのローカル識別は、 **/etc/passwd** や **/etc/group** ファイル内のエントリーなどの、ローカル・マシン上にあるローカル・データにより定義されています。 **passwd_export** コマンドを使用して、 **/etc/passwd** ファイルや **/etc/group** ファイルを、ローカル・レジストリー内に格納されているデータでもって更新する場合、ローカル識別データは、ネットワーク・レジストリーに格納されている情報から得られます。

レジストリー・アカウントは、プリンシパルを、グループ、オーガニゼーション、および、関連するアカウント情報 (プリンシパルの識別を認証するために使用されるパスワードなど) に関連付けることにより、ネットワーク識別を定義します。通信が認証されるかどうかに関係なく、ネットワークにおいて通信にかかわるすべてのプリンシパルに対して、レジストリー・アカウントを作成する必要があります。レジストリー・アカウントを作成する必要があるプリンシパルは、以下のものです。

- ネットワーク上でオブジェクトにアクセスする、それぞれの人間のユーザー。特にユーザーがローカル・マシンに制限されない限り、これにはすべての人間のユーザーが含まれます。
- ネットワークを通してオブジェクトにアクセスし、サーバーを開始したプリンシパルの識別でなく、自身の識別の下で稼働する、それぞれのサーバー。
- ネットワーク内の各マシン。
- 認証されたセル間通信を行うセル。 (セル間認証のアカウントは、431ページの『第33章 マルチセル環境の管理』で説明されている、特別のタイプのアカウントです。)

ここでは、次のことについて説明します。

- 各タイプのアカウントおよびそれを作成し、保守する方法
- アカウントが認証される方法および特権属性とチケットを表示する方法
- サーバー・プリンシパルのキーを保管する keytab ファイルを作成し、保守する方法
- ローカル・レジストリーを保守する方法

ユーザー・アカウント

ユーザー・アカウントは、ユーザーが DCE にログインするときに使用する、ユーザーのパスワードと情報に関連しています。アカウント情報には、プリンシパルのホーム・ディレクトリーとログイン・シェル、およびプリンシパルの DCE へのアクセスの制御を支援するパラメーターを定義する、認証ポリシーが含まれます。人間のユーザーのアカウントを作成するには、 **dcecp account create** コマンドを使

用し、それらを変更するには **dcecp account modify** コマンドを使用し、またそれらを削除するには、**dcecp account delete** コマンドを使用します。

サーバー・アカウント

サーバー (これはアプリケーション とも呼ばれます) は、ネットワークを通して通信し、自身のネットワーク識別 、あるいはサーバーを開始したプリンシパルのネットワーク識別の下で、実行することができます。サーバーが自身の識別のもとで実行するには、サーバーがログインを行い、その識別を認証するように、プログラミングする必要があります。したがって、**dcecp account create** コマンドを使用して、これらのサーバーのレジストリー・アカウントを作成する必要があります。

サーバー・アカウントのパスワード

ログインの際に、すべてのプリンシパル (人間、サーバー、およびマシン) は、パスワードを DCE 認証サービスに渡さなければなりません。認証サービスは、これらのパスワードを使用して認証キーを生成します。人間のユーザーの場合、最も一般的な方法はパスワードを単純に入力することです。サーバー・プリンシパルについては、別の方法を提供する必要があります。推奨される方法 (DCE により提供される API に基づく) は、ローカルに保護されたキー・テーブルにサーバー・キーを格納する方法です。 DCE 提供の API のデフォルトでは、キー・テーブルをサーバーのローカル・マシン上の keytab ファイルに格納し、そして、このファイルを保護して、プリンシパルのローカル識別のみがこのファイルを読み書きできるようにします。

keytab ファイルは、リモートからアクセスすることができます。ローカル・マシン上では、この keytab ファイルは、どのようなファイル・システムによってもエクスポートされない、マシンのディスクのパーティションに格納してください。

ルートとして稼働しているか、またはローカル・マシンの識別のもとで稼働しているサーバーを除き、それぞれのサーバーに対して個別の keytab ファイルを使用する必要があります。ログインのときに、サーバーはこのファイルにアクセスしてキーを入手し、そのキーを認証サービスに渡し、ログインして、そして認証を受けます。

keytab ファイルにサーバーのキーを追加するには、**dcecp keytab add** コマンドを使用し、サーバー・キーを削除するには、**dcecp keytab remove** コマンドを使用します。

サーバー・アカウントを作成するステップ

サーバーのアカウントを作成するには、まず最初に、**dcecp account create** コマンドを実行してアカウントを作成し、次に、**dcecp keytab add** コマンドを実行してエントリーを keytab ファイルに追加します。レジストリー内のサーバーのパスワードと keytab ファイル内のサーバーのキーは、一致しなければなりません。両方のコマンドに同じパスワードを手作業で入力することによってこれらのパスワードを同じにするか、あるいは keytab ファイルにサーバーのパスワードをセットするときに、同時にサーバーのレジストリー・パスワードをリセットするように、 **keytab add** コマンドに指定することができます。

マシン・アカウント

すべてのマシンは、レジストリーにアカウントを持っていなければなりません。マシン・アカウントは、サーバー・アカウントと同様に、まず最初に、**account create** コマンドを実行してアカウントを作成し、次に、**keytab add** コマンドを実行して、サーバーのパスワードを keytab ファイルに追加します。サーバー・アカウントと同様に、レジストリー内のマシン・アカウントのパスワードと keytab ファイルのパスワードは一致しなければなりません。マシン・アカウント内のプリンシパル名は、セル・ネームスペース内のマシンの名前と同じでなければなりません。(セル・ネームスペース内の名前の詳細については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド － 入門* を参照してください。)

アカウントが表す識別を認証する方法

プリンシパルが DCE にログインするときに、セキュリティー・クライアントはプリンシパルが提供する (あるいは、サーバー・プリンシパルまたはマシン・プリンシパルの場合は、それらに提供される) パスワードを使用して、プリンシパルの認証キーを引き出します。プリンシパルの認証キーのコピーは、レジストリー・データベースにも存在します。これは、プリンシパルのアカウントが作成されたときに (あるいはパスワードが変更されたとき)、ここに格納されたものです。したがって、認証キーを認証サービスが使用できます。

このキーは、認証サービスがプリンシパルを認証するため (つまり、プリンシパルの識別を保証するため) に次のように使用します。

1. セキュリティー・クライアントは以下のことを行います。
   a. ユーザーにパスワードを照会し、それを使用してプリンシパルの認証キーを引き出す。
   b. ログイン要求を作成し、その要求の一部を認証キーを使用して暗号化する。
   c. 要求を認証サービスに転送する。
2. 認証サービスは以下のことを行います。
   a. ログイン要求を受け取る。
   b. プリンシパルの認証キーのレジストリーのコピーを入手する。
   c. このキーを使用して、ログイン要求の復号を試みる。

複号が成功すると、キーは同じになります。プリンシパルは認証され、ログインは成功します。

復号が失敗した場合、プリンシパルが提供し、セキュリティー・クライアントがプリンシパルの認証キーを引き出すために使用したパスワードは無効であり (つまり、プリンシパルの認証キーのレジストリーのコピーを引き出すために使用されたパスワードと異なる)、ログインは拒否されます。

上に述べたものは非常に一般的な概要にすぎません。プリンシパル認証の詳細については、*IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* を参照してください。

特権属性

プリンシパルが認証された後、 DCE セキュリティー・サービスを使用してプリンシパルの特権属性を入手することができます。特権属性は、プリンシパルのネットワーク識別を表す UUID、プリンシパルがメンバーであるグループ、およびプリンシパルに関係するすべての拡張属性によって構成されています。特権属性は、プリンシパルがオブジェクトへのアクセスを要求する際に、そのオブジェクトに対する権利を判別するために使用されます。 DCE セキュリティー・サービスによって提供される特権属性は、認証されます。認証された特権は、ネットワーク・サービスにより受け入れられます。認証されない特権属性は、受け入れられません。これは、プリンシパルに許される DCE オブジェクトへのアクセスの種類が、プリンシパルの特権属性が認証されるかどうかによって異なることを意味します。 (DCE ACL は、プリンシパルの特権属性に基づいて DCE オブジェクトへのアクセスを制御するために使用されるもので、これについては、329ページの『第28章 アクセス制御リストの使用』に記述されています。)

チケット発行チケットおよびサービスへのチケット

チケット発行チケットは、たとえば、プリンシパルが、ファイルを読むために、分散ファイル・システム・サーバーなどの DCE サービスへのチケットを要求したり、受け取ったりできるようにします。プリンシパルに DCE サービスをアクセスさせるチケットは、*サービス・チケット* と呼ばれます。

チケット発行チケットとサービス・チケットはいずれも、個々のアカウントおよびレジストリー・ポリシーと属性の設定によって決まる、存続時間を持っています。プリンシパルのチケット発行チケットの有効期限が切れると、プリンシパルは、もはや認証されたユーザーとは見なされません。認証されていないプリンシパルの、ローカル・マシンのオブジェクト以外のオブジェクトに対するアクセスは厳しく制限されており、そのプリンシパルが DCE サービスを使用する能力は極度に制限されます。これを矯正するには、プリンシパルは **kinit** コマンドを実行することによって再認証するか (*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照)、あるいは DCE からログアウトし、再度ログインすることによって再認証する必要があります。

プリンシパルのサービス・チケットは認証サービスによって自動的に更新され、プリンシパル側でのアクションは必要ありません。ただし、サービス・チケットに割り振られる存続時間は、プリンシパルのチケット発行チケット (TGT) の残り時間を超えることはできません。

特権属性およびチケットの表示

DCE セル管理者は、**klist** コマンドを使用して、プリンシパルの現在のチケットと特権属性を表示することができます。 **klist** コマンドは、特権属性、有効期限情報、およびサービス・チケット情報の、 3 種類の情報を表示します。 DCE ユーザーは、**klist** を実行して、自分の現在のチケットと期限切れのチケットを表示することができます。 **klist** コマンドについては、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* で説明されています。

klist 表示の最初の部分 - 特権属性

klist コマンドは、プリンシパルの特権属性を表示します。この表示は、最初にプリンシパルの完全修飾名をリストし、続いてセルの UUID と名前、プリンシパル名

(セル名と DCE グローバル ID はない)、およびプリンシパルがメンバーであるすべてのグループをリストします。 **klist** 表示のこのセクションの例を次に示します。

```
DCE Identity Information:
   Global Principal: /.../dresden.com/music/mozart
   Cell: 5ad96550-80c4-11ca-b26c-08001e039431 /.../dresden.com
   Principal: 00000066-80c5-11ca-b600-08001e039431 music/mozart
   Group: 00000003-80c4-11ca-b201-08001e039431 composers
```

klist 表示の 2 番目の部分 - 有効期限の日付と時刻

klist 表示の 2 番目の部分は、プリンシパルのチケット発行チケット、アカウント、およびパスワードの期限が切れる日時を示します。

- 最初の行は、チケット発行チケットの期限が切れる日時を示します。この期限が切れる前に、プリンシパルは **kinit** を実行するか、 DCE に再度ログインして、これを再初期化する必要があります。
- 2 番目の行は、プリンシパルのアカウントの有効期限を示します。アカウントの期限が切れると、プリンシパルは DCE にログインすることができなくなります。これを矯正するには、 DCE 管理者は、レジストリー内のプリンシパルのアカウント有効期限日を変更しなければなりません。
- 3 番目の行は、プリンシパルのパスワードの有効期限の日付を示します。この期限が切れる前に、プリンシパルは **dcecp** を使用して、パスワードを変更する必要があります。パスワードが失効すると、プリンシパルは DCE にログインすることができなくなります。これを矯正するには、 DCE 管理者は、レジストリー内のプリンシパルのパスワードを変更しなければなりません。

klist 表示の 2 番目の部分の例を次に示します。

```
Identity Info Expires: 91/10/03:12:07:18
Account Expires: 91/12/31:12:00:00
Passwd Expires: 91/10/31:12:00:00
```

klist 表示の 3 番目の部分 - チケット

klist 表示の 3 番目で、かつ最後の部分は、プリンシパルのチケット情報とプリンシパルのチケット・キャッシュの名前を示します。次の表示の **Server** のラベルの付いた最初の 3 つのチケットは、プリンシパルがログインし、特権属性を入手した後に使用されたチケットです。すべてのプリンシパルの表示には、これらのエントリーがあります。

Client のラベルが付いた残りのチケットは、プリンシパルのチケット発行チケットとサービス・チケットを示します。 **Client** という単語の後の各チケットのリストでは、表示は特権サーバーの名前を示します。特権サーバーとは、プリンシパルの識別が DCE セキュリティー・サービスにより認証された後に特権属性を与えるサーバーです。プリンシパルが持っているチケットの相手のサーバーの名前は、 **Server** エントリーの後に示されており、これらのチケットが有効である日時は、その次の行に示されています。たとえば、次の表示の例では、最後の行は、プリンシパルが **file_server** という名前のサーバーへのチケットを持っていることを示しています。このチケットの存続時間は、 91 年 10 月 2 日の午後 1 時 24 分 2 秒から 91 年 10 月 3 日の午後 12 時 7 分 18 秒までです。 (時間は、24 時間形式で示されます。)

```
Kerberos Ticket Information:
Ticket cache: /tmp/dcecred_17a80000
Default principal: music/mozart@dresden.com
```

```
Server: krbtgt/dresden@dresden.com
   valid 91/10/02:12:07:18 to 91/10/03:12:07:18
Server:dce/rgy@dresden.com
   valid 91/10/02:12:07:20 to 91/10/03:12:07:18
Server:dce/ptgt@dresden.com
   valid 91/10/02:12:07:49 to 91/10/03:12:07:18
Client:dce/ptgt@dresden       Server:krbtgt/dresden@dresden.com
   valid 91/10/02:12:07:50 to 91/10/03:12:07:18
Client:dce/ptgt@dresden.com   Server:dce/rgy@dresden.com
   valid 91/10/02:12:07:53 to 91/10/03:12:07:18
Client:dce/ptgt@dresden.com   Server:file_server@dresden.com
   valid 91/10/02:13:24:02 to 91/10/03:12:07:18
```

プリンシパルのチケットの破棄

プリンシパルが獲得したチケットを無効にするには、**kdestroy** コマンドを使用します。プリンシパルがログアウトするとき、プリンシパルのチケットは破棄されません。それらは、有効期限が切れるまで有効です。 DCE ユーザーは、ログアウトする直前に **kdestroy** を使用して、有効なチケットが残らないようにすることができます。ただし、プリンシパルにカーネル常駐のチケット・キャッシュがある場合は、そのプリンシパルの最後のプロセスが終了した時点で、プリンシパルのチケットが破棄されます。これは、通常はログアウト時に **kdestroy** を実行する必要がないことを意味します。

kdestroy コマンドについては、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* で説明されています。

アカウントの追加

アカウントをレジストリーに追加するには、 **dcecp account create** コマンドを使用します。アカウントに関係する情報は、おおまかに次の 2 つのカテゴリーに分けられます。

- これに類似したユーザー情報は通常、 **/etc/passwd** ファイルにあります。
- アカウントのネットワークへのアクセスを制御できるようにする認証ポリシー。認証ポリシーは、アカウントとパスワードの妥当性、アカウント有効期限ポリシー、およびチケット有効期限ポリシーを確立します。認証ポリシーを厳格にすればするほど、セルは安全になりますが、処理のオーバーヘッドが増大します。

両方のタイプの情報は、標準の **dcecp** 属性リスト内の属性として、あるいは属性オプションとして提供されます。

レジストリーには、認証ポリシーも設定できることに注意してください。レジストリーのポリシーがアカウント用に入力されたポリシーと異なると、より厳格なポリシーが適用されます。 (異なるポリシーについての詳細は、 447ページの『第35章 ポリシーと属性の保守』を参照してください。)

401ページの表22 は、アカウントを作成するために使用される、属性オプションをリストしています。この表で説明されているオプションは、属性リストの中にダッシュを使用しないで指定できることに注意してください。

表 22. アカウントを作成するための属性オプション

オプション	意味
-acctvalid {yes|no}	アカウントの妥当性を決めるフラグ。このフラグを **no** にセットすると、アカウントは無効となり、アカウント・プリンシパルはアカウントにログインできません。デフォルトは、**yes** です。
-client {yes|no}	アカウントがクライアントとして作動できるプリンシパルに対するものであるかどうかを示すフラグ。このフラグを **yes** にセットすると、プリンシパルはアカウントにログインして、認証のためのチケットを得ることができます。デフォルトは、**yes** です。
-description *string*	通常、アカウントの用途を記述するために使用される、ポータブル文字セット (PCS) 形式のテキスト・ストリング。デフォルトはありません。
-dupkey {yes|no}	アカウントのプリンシパルに発行されたチケットが複写キーを持つことができるかどうかを決めるフラグ。デフォルトは **no** です。
-expdate	アカウントの有効期限が切れる日付 (ISO タイム・スタンプ形式 *YY-MM-DD-hh:mm:ss*)。アカウントの有効期限が切れたあと、アカウントを更新するには、日付を変更します。デフォルトは **none** です。これはアカウントの有効期限が決して切れないことを意味します。
-forwardabletkt {yes|no}	現在のチケット発行チケット (TGT) のネットワーク・アドレスと異なるネットワーク・アドレスを持つ新しいチケット発行チケットを、アカウントのプリンシパルに発行することができるかどうかを決めるフラグ。 (**-proxiabletkt** 属性は、サービス・チケットに対して同じ機能を実行します。) デフォルトは、**yes** です。
-goodsince *date*	アカウントが最後に非汚染状態になった日時 (ISO タイム・スタンプ形式 *YY-MM-DD-hh:mm:ss*)。この日付以前に与えられたチケットは無効です。アカウントのパスワードが汚されていることが分かっている場合、この日付の制御は特に役に立ちます。パスワードを変更することにより、許可されていないプリンシパルがそのパスワードを使用して再度システムにアクセスできないようにすることができます。ただし、そのプリンシパルがパスワードの変更前に不正に入手したチケットを用いてシステム・コンポーネントにアクセスすることは防止できません。プリンシパルのシステムへのアクセスを除去するには、チケットを取り消さなければなりません。 **-goodsince** 属性を汚染されたパスワードが変更された日時にセットしてください。そうすれば、このときより前に発行されたチケットはすべて無効になり、許可されていないプリンシパルがシステムにアクセスできなくなります。アカウントが作成されるとき、**-goodsince** 属性は現在の日付にセットされます。

表 22. アカウントを作成するための属性オプション (続き)

オプション	意味
-group *group_name*	アカウントに関連するグループの名前。アカウントを作成するには、この属性を指定しなければなりません。デフォルトはありません。
-home *dir_name*	ログイン時にプリンシパルが置かれるディレクトリー。デフォルトはありません。
-organization *org_name*	アカウントに関連するオーガニゼーションの名前。アカウントを作成するには、この属性を指定しなければなりません。デフォルトはありません。
-password *password*	アカウントに必要な、プレーン・テキストのパスワード。システムは、入力されたパスワードを暗号化します。デフォルトはありません。
-postdatedtkt {yes|no}	未来の開始時間を持つチケットを、そのアカウントのプリンシパルに対して発行できるかどうかを決めるフラグ。デフォルトは **no** です。
-proxiabletkt {yes|no}	現在のチケットとは異なるネットワーク・アドレスを持つ新しいチケットを、そのアカウントのプリンシパルに対して発行できるかどうかを決めるフラグ。(**-forwardabletkt** 属性オプションは、チケット発行チケットに対して同じ機能を実行します。) デフォルトは **no** です。
-pwdvalid {yes|no}	現在のパスワードが有効であるかどうかを決めるフラグ。このフラグを **no** にセットすると、アカウント・パスワードは期限切れとなり、プリンシパルは、次回にアカウントにログインするときに、そのパスワードを変更するようにプロンプトが出されます。デフォルトは、**yes** です。
-renewabletkt {yes|no}	Kerberos V5 更新可能チケット機能は、現在、DCE では使用されていません。この更新可能チケット属性の使用は、現在サポートされていません。
-server {yes|no}	アカウントがサーバーとして活動することができるプリンシパルのためのものであるかどうかを示すフラグ。アカウントが、認証通信を行うサーバーのためのものである場合は、このフラグを **yes** にセットします。デフォルトは、**yes** です。
-shell *path_to_shell*	プリンシパルのログイン時に実行されるシェル。
-stdtgtauth {yes|no}	アカウントのプリンシパルに発行されたチケットが、チケット発行チケット認証メカニズムを使用できるかどうかを決めるフラグ。デフォルトは、**yes** です。
-usertouser {yes|no}	サーバー・プリンシパルの場合、サーバーがユーザー間認証を使用すべきか否かを決めるフラグ。値は **yes** (ユーザー間プロトコルに基づいた認証を使用すべき)、または **no** (サーバーとキーのチケット・プロトコルに基づいた認証を使用する)。デフォルトは **no** です。

表 22. アカウントを作成するための属性オプション (続き)

オプション	意味
-maxtktlife *hours*	最大チケット存続時間。これは、チケットが有効である最大の長さ (時間) です。クライアントがサーバーへのチケットを要求すると、そのチケットに与えられる存続時間は、サーバーとクライアントの両方の **maxtktlife** 属性値が考慮されます。言いかえると、存続時間は、サーバーあるいはクライアントの最大チケット存続時間のどちらか短い方の値を超えることができません。 アカウントに **maxtktlifetime** 属性値を指定しないと、レジストリーの認証ポリシーに定義された **maxtktlifetime** 属性値が使用されます。 (『ポリシーと属性の保守』を参照してください。)
-maxtktrenew *hours*	最大チケット更新時間。これは、プリンシパルのチケット発行チケットが期限切れとなり、プリンシパルがシステムに再度ログインし、再度認証を行い、新しいチケット発行チケットを入手しなければならないときまでの時間数です。 プリンシパルのサービス・チケットの存続時間は、プリンシパルのチケット発行チケットの存続時間を超えることはできません。最大チケット更新時間を短くするほど、システムのセキュリティーの度合いは高まります。ただし、プリンシパルは再度ログインしてチケット発行チケットを更新しなければならないため、ユーザーの便利さと必要なセキュリティーのレベルを考慮に入れて時間を決める必要があります。 アカウントに **maxtktrenew** 属性値を指定しないと、レジストリーの認証ポリシーに定義された **maxtktrenew** 属性値が使用されます。 (『ポリシーと属性の保守』を参照してください。) DCE RPC は現在、サービス・チケットの更新時に、更新可能チケットの機能を使用していません。しかし、DCE セキュリティー・サーバーはこれをサポートしており、DCE セキュリティー・サーバーを KDC として使用する Kerberos V5 アプリケーションにとってはこの機能は便利です。

注: 最大チケット存続時間と最大チケット更新時間は、 **dcecp registry modify** コマンドにより、レジストリーのレジストリー属性としてまとめてセットすることができます。 **dcecp account create** あるいは **account modify** コマンドを使用してセットされた場合、それらは特定のアカウントだけに適用されます。

チケット存続時間のセット

dcecp registry modify コマンドでセットされる、他の 2 つのオプション、すなわち、デフォルトのチケット存続時間と最小チケット存続時間、についても注意する必要があります。

- 最小チケット存続時間 - チケットに割り当てることが可能な最小の存続時間。最小チケット存続時間の実際の有効な値は、デフォルトの証明存続時間に影響されることに注意してください。
- デフォルトのチケット存続時間 - プリンシパルが特に異なる存続時間を要求しない限り、そのチケットに対して与えられる存続時間。プリンシパルは、チケットについて特定の存続時間を要求することができますが、大多数は、デフォルトの存続時間を受け入れます。 (プリンシパルが 0 (ゼロ) のチケット存続時間を要求した場合は、デフォルトの存続時間がチケットに対して割り当てられます。)

  デフォルトのチケット存続時間の実際の有効な値は、最大証明存続時間に影響されることに注意してください。

チケットに割り当てられる実際の存続時間は、最大チケット存続時間、デフォルトのチケット存続時間、および最小チケット存続時間の設定に関して DCE セキュリティー・サービスが使用する、規則によって決まります。それらの規則は、次のとおりです。

- 最大チケット存続時間は、更新可能チケット存続時間より大きくすることはできず (つまり、 **max_life = min (max_life, renewable_life)** であり)、また 60 秒より小さくすることもできません。最大チケット存続時間が更新可能チケット存続時間よりも大きいと、更新可能チケット存続時間が最大チケット存続時間として使用されます。たとえば、あるアカウントが 15 時間にセットされていると仮定します。更新可能チケット存続時間を 20 時間にセットすると、有効な最大チケット存続時間は、20 時間ではなく、15 時間となります。
- デフォルトのチケット存続時間は、最大チケット存続時間よりも大きくすることはできず (つまり、 **default_life = min (default_life, max_life)** であり)、また 60 秒より小さくすることもできません。デフォルト・チケット存続時間が最大チケット存続時間よりも大きいと、最大チケット存続時間がデフォルト・チケット存続時間として使用されます。たとえば、レジストリー・ポリシーがデフォルト・チケット存続時間として 25 時間を指定しているとします。レジストリーの最大チケット存続時間を 15 時間に設定すると、レジストリーの有効なデフォルト証明存続時間は、 25 時間ではなく、15 時間となります。
- 最小チケット存続時間は、デフォルト証明存続時間より大きくすることはできず (つまり、 **min_life = min (min_life, default_life)** であり)、また、60 秒より小さくすることもできません。最小チケット存続時間が、デフォルト証明存続時間よりも大きいと、デフォルト・チケット存続時間が最小チケット存続時間として使用されます。たとえば、レジストリー・ポリシーがデフォルト・チケット存続時間として 10 時間を指定しているとします。アカウントの最小チケット存続時間を 15 時間に設定すると、そのアカウントの有効最小チケット存続時間は、 15 時間ではなく、10 時間となります。

dcecp では規則に反する値を入力することが可能であり、アカウントのポリシーを表示すれば (**account show** コマンドを使用) これらの値を見ることができますが、使用される値は規則で記述されている値であり、入力された値ではありません。

注: 厳密にいうと、ネットワークのクロックは、レジストリー・データと関連している時間と同期していなければなりません。

チケット発行チケット存続時間とサービス・チケット存続時間

認証サービスは、プリンシパルのチケット発行チケット存続時間の残り時間を超える存続時間を持つサービス・チケットをプリンシパルに与えることはしません。たとえば、プリンシパルのチケット発行チケットの存続時間として 2 時間が残っているときに、プリンシパルがサービス・チケットの存続時間として、 4 時間のデフォルトを要求あるいは受け入れる場合、2 時間の存続時間しか与えられません。

更新可能チケット・フラグ (**renewabletkt** 属性) がプリンシパルのアカウントに対してオンにセットされている場合、プリンシパルのチケット発行チケットの存続時間は、サービス・チケットの更新にも影響します。サービス・チケットが、プリンシパルのチケット発行チケットの残りの存続時間を超える存続時間を持って更新されることはありません。サービス・チケットは、通常は、元のチケットに割り振られた存続時間に合わせて更新されます。元の時間がチケット発行チケットの存続時間を超えている場合、そのチケットは、チケット発行チケットの残りの時間だけ更新されます。

アカウントを追加する例

アカウントを作成するには、**dcecp account create** コマンドを使用します。**account create** コマンドを使用するとき、アカウントを作成しようとしているプリンシパルの名前および、そのアカウントが関連するグループとオーガニゼーションの名前を入力する必要があります。さらに、このコマンドを入力する人は、自分の識別を確認させるために、 **-mypwd** オプションとともに自分のパスワードを入力する必要があります。パスワードを入力しないと、**dcecp** はパスワードを入力するようプロンプトを出します。その他の属性はすべてデフォルトとすることができます。パスワードのプロンプトが出された場合、入力するパスワードは画面に表示されないので注意してください。

account create コマンドの使用者はパスワードを入力する必要があるため、このコマンドは対話モードで実行する必要があります。パスワードを画面に表示せざるをえないコマンド行モードでは、このコマンドを実行することができません。

次の例は、**dcecp account create** コマンドを使用して、グループ **symphonists** とオーガニゼーション **classic** に関連するプリンシパル **mahler** のアカウントを作成するものです。その他のアカウント属性は、デフォルトとすることができます。

```
dcecp> account
create mahler -group symphonists -organization classic ¥
> -password passwd -mypwd cellpwd
dcecp>
```

1 つの **account create** コマンドで複数のアカウントを作成することが可能です。このためには、アカウントが作成されるプリンシパルの名前を中括弧で囲み、それぞれをスペースで区切ります。たとえば、プリンシパル **bach**、 **britten**、および **mahler** のアカウントを作成するには、次のように入力することができます。

```
dcecp> account create
{bach britten mahler} -group symphonists ¥
> -organization classic -password music -mypwd cellpwd
dcecp>
```

複数のアカウントを作成する場合、各アカウントには同じ属性が割り当てられます。これは、この例においては、**bach**、**britten**、および **mahler** のアカウントはすべて、 **symphonists** グループと **classic** オーガニゼーションに関連付けられ、すべてにパスワード **music** が割り当てられることを意味します。すべてが同じグループとオーガニゼーションに属するプリンシパル用の複数のアカウントは、このように作成するのが便利です。アカウントがこのように作成されたユーザーに対しては、直ちにパスワードを変更するように通知してください。

アカウントの変更

dcecp account modify コマンドにより、アカウントを変更することができます。すべてのアカウント属性を変更することができます。

アカウントを変更するときには、アカウント変更を行う人の識別を確認させるために、 **-mypwd** オプションとともに自分のパスワードを入力する必要があります。パスワードのプロンプトが出された場合、入力するパスワードは画面に表示されないので注意してください。パスワードを入力しないと、パスワードを入力するようにプロンプトが出されます。 **account modify** コマンドの使用者はパスワードを入力する必要があるので、このコマンドは対話モードで実行する必要があります。パスワードを画面に表示せざるをえないコマンド行モードでは、このコマンドを実行することができません。

ユーザーのアカウントの **maxtktrenew** または **maxtktlife** 属性を設定するために、**account modify** コマンドを使用すると、次のエラーを受け取ることがあります。

```
msgID=0x17122084 データ・レコードが無効です。
```

これら 2 つのアカウント属性のどちらでも初めて設定する場合には、たとえこれらの属性に有効な値を指定するとしてもこれは生じます。アカウントの **maxtktrenew** または **maxtktlife** 属性を初めて設定する場合は、両方の属性を dcecp **account modify** コマンドで指定する必要があります。アカウントに両方の属性を設定し終えたら、 dcecp を使用して両方の属性を個別に変更できます。

次の例は、**account modify** コマンドを使用して、 **mahler** のアカウントに新しいホーム・ディレクトリーを指定する方法を示しています。

```
dcecp> account modify mahler -home /.../music/fs/users/mahler/concert
¥
> -mypwd cellpwd
dcecp>
```

属性リストの変更は、**account modify** に **-change** オプションを使用してもできます。 **-add** および **-remove** オプションは、 **account modify** コマンドではサポートされていません。それは、各アカウント属性が存在していなければならず、また値を持っていなければならないためです。

アカウントの削除

次の例は、**dcecp account delete** コマンドを使用してプリンシパル **mahler** のアカウントを削除する方法を示しています。

```
dcecp> account delete  mahler
dcecp>
```

グループあるいはオーガニゼーションを削除すると、グループあるいはオーガニゼーションに関連するアカウントも自動的に削除されます。

1 つの **account delete** コマンドで複数のアカウントを削除することができます。このためには、アカウントのプリンシパルの名前を中括弧で囲み、それぞれをスペースで区切ります。たとえば、アカウント **bach**、**britten**、および **mahler** を削除するには、次のように入力します。

```
dcecp> account
delete {bach britten mahler}
dcecp>
```

keytab ファイルの作成、保守、および削除

次の **dcecp** コマンドにより、 keytab ファイルを作成、保守、および削除することができます。

keytab create
: keytab ファイルと、そのすべてのキー・エントリーを作成します。

keytab delete
: keytab ファイルと、そのすべてのキー・エントリーを削除します。

keytab add
: keytab ファイルにキー・エントリーを追加します。

keytab remove
: keytab ファイルからキー・エントリーを削除します。

次に keytab ファイルを管理する方法について説明します。

Keytab ファイル

keytab ファイルにはサーバーのキーが含まれており、この keytab ファイルはサーバーと同じマシンに格納されます。それらは、**dcecp** を使用して、リモートからまたローカルでアクセスすることができます。リモート・アクセスの場合、**dcecp** は **dced** インターフェースを使用します。 **-local** オプションを **dcecp keytab** コマンドに指定すると、 **dced** を使用せずに、ローカル keytab ファイルをアクセスすることができます。

dced は、keytab ファイルへのリモート・アクセスを提供するので、これらのファイルは **dced** オブジェクトとして定義され、またこれらのオブジェクトは **keytab** ディレクトリーのもとでネームスペースの **dced** 制御の部分に格納されます。 **dced** keytab オブジェクトは、オブジェクトを識別する UUID、オプショナルの注釈、およびローカル・マシンにサーバー・キーを実際に格納するファイルの名前で構成されます。通常、このオブジェクトはファイルです。

実際のサーバー・キーは、keytab オブジェクトに格納されるのではなく、ローカル・マシン上のファイルに格納されることに注意してください。

dced keytab オブジェクトのパス名は、次のとおりです。

/.:/hosts/*hostname***/config/keytab/***keytab_name*

ここで、

hostname
これは、**dced** プロセスが常駐するホストの名前です。

keytab_name
これは、keytab ファイルの名前です。

ローカル keytab ファイルへのパス名は以下のとおりです。

/opt/*dcelocal*/*keytab_path_name*

ここで、

keytab_path_name
これは、ローカル・ノード上の keytab ファイルへのパス名です。

Keytab ファイルの保護

ローカル keytab ファイルは、適切に保護する必要があります。またそれらは、ネットワークで使用できないようにする必要があります。デフォルトの DCE 実装で使用される時に、 keytab ファイルには、DCE セキュリティーの基本である、プリンシパルのキーが含まれています。これらのキーが汚されると、ネットワークのセキュリティーも汚されます。 keytab ファイルをアクセスする呼び出しは、**rpc_c_protect_level_pkt_privacy** を使用します。この保護レベルは、渡すデータにデータ暗号化規格 (DES) 暗号化を行います。この保護レベルでなく、自分のサイトのデフォルト保護レベルを使用する必要がある場合は、 **dcecp keytab -noprivacy** オプションを指定します。

各ローカル・ノードで実行するサーバー・プリンシパルごとに、個別の keytab ファイルを作成します。同じ keytab ファイルを共用するサーバーは、互いのキーにアクセスできるため、互いに相手に偽名を使用することができます。 keytab ファイルをルートだけが読み取れるように保護してください。このようにした場合、サーバーは、ログインのときに keytab ファイルを読み取り、キーを入手するために、ルートにより開始されなければなりません。

サーバー・キーを作成または変更するとき、ローカル・ノードで実行されるサーバーごとに異なる keytab ファイルを指定できます。ファイルに自身のキーが含まれているサーバーだけがそのファイルを読み取ることができるように、ファイルを保護してください。それから、サーバー・ファイルの **setuid** ビットをサーバーの識別にセットして、サーバーが keytab ファイルをアクセスし、そのキーを入手できるようにします。

サーバー・キーおよびマシン・キーのバージョン番号

キーが keytab ファイルに追加されるとき、それぞれのキーには 1 から 255 までのバージョン番号が割り当てられます。サーバーまたはマシンのキーが変わる (自動的にあるいは明示的に) たびに、キーのバージョン番号が増やされます。バージョン番号により、特定のサーバーあるいはマシンに複数のキーが存在できるようになります。キーが変更されたとき、そのキーの古い変更されないバージョンのもとで与えられたチケットをまだ使用しているサーバーあるいはマシンは、そのチケットが自然に期限切れとなるまで、中断されることなく稼働します。チケットが期限切れになると、サーバーあるいはマシンは再度認証し、新しいキーを入手します。

-registry オプションを **keytab add** コマンドに使用すると、可能であれば、古いキーは自動的に削除されます。このオプションを使用しない場合は、 **keytab list**

コマンドを使用して keytab ファイルの内容を時々リストし、そして **keytab delete** コマンドを使用して、使用されなくなった古いバージョンを削除する必要があります。

注: keytab ファイルからキーを削除するときには、注意する必要があります。プリンシパルのキーが変更されるとき、削除されたキーに基づいたチケットがまだ残っていることがあります。 keytab ファイルからキーを削除すると、削除されたキーに基づいた活動状態のチケットはサーバーによって受け入れられず、これらのチケットを保持するクライアントは認証に失敗します。

キーと Keytab ファイルの作成および保守

次の 2 つのコマンドを使用して、キー・エントリーを作成することができます。

keytab create
これは keytab ファイル、keytab ファイルのエントリー、および **dced** keytab オブジェクトを作成します。

keytab add
これは既存の keytab ファイルにキー・エントリーを追加します。

この両コマンドを実行する際には、作成または変更する keytab ファイルの名前を指定します。

表23 に、**keytab create** および **add** コマンドに指定できる、その他のオプションがリストされています。

表 23. keytab 作成と keytab 追加のオプション

オプション	意味
-local	**dced** を使用せずに、keytab ファイルをアクセスします。
-entry	実際のキー・テーブルでなく、**dced** 構成情報だけを作成します。
-noprivacy	このオプションは、保護レベルとして **rpc_c_protect_level_pkt_privacy** でなく、ユーザーのサイトのデフォルト保護レベルを使用することを指定します。
-member *name*	キーを作成あるいは変更するプリンシパル (サーバーあるいはマシン) の名前。リストには、複数の名前を指定することができます。リストで与える場合、リストに指名されているすべてのプリンシパルに同じキーが割り当てられます。
-key *key*	アカウントへのプレーン・テキスト・キー。このオプションは、 **-random** オプションとともに使用することはできません。
-random	ランダム・キーを生成します。このオプションを使用する場合、**-registry** オプションも使用して、ランダムに生成されたキーをレジストリー内のサーバーまたはマシンのアカウントに追加する必要があります。このオプションは、**-key** オプションとともに使用することはできません。

表 23. keytab 作成と keytab 追加のオプション (続き)

オプション	意味
-registry	このオプションは、keytab ファイルのキーとして入力した (あるいは自動的に生成された) キーに一致するように、レジストリー内のプリンシパルのキーを更新します。これを使用すると、keytab ファイルのプリンシパルのキーを変更するときに、レジストリーと keytab ファイルの中のプリンシパルのキーが同期します。 このオプションは、 **-random** オプションを使用する場合は必須です。このオプションを使用する場合、**dcecp login** コマンドを実行して、ユーザーのネットワーク識別がレジストリー・データベースの変更に適切なものであることを確認する必要があります。ユーザーが **-registry** オプションを使用する場合、プリンシパルのパスワード (キー) がキー・テーブル・ファイルに存在しなければなりません。
-version *number*	これは、キーのバージョン番号を指定します。これは、**-registry** オプションを使用しない場合に必要です。
-storage *local_file_name*	作成するローカル・ファイルのパス名。このオプションは、**keytab create** コマンドに対してのみ使用されます。既存の keytab ファイルにエントリーを追加するとき、 **dced** オブジェクト名によってファイルを識別します。
-data *keys*	次に示す形式の、サーバー・プリンシパル名およびキー。 *principal_name key_type* { *version*} {*key_value*}

keytab ファイルの作成

keytab ファイル、ファイルのエントリー、および対応する **dced** オブジェクトを作成するには、 **keytab create** コマンドを使用します。このコマンドを使用する場合は、引き数として、作成する **dced** オブジェクトのパス名、 keytab のローカルを指定する **storage** オプション、サーバー・プリンシパルとキーの名前を指定する **data** オプション、および 409ページの表23 にリストされている適切なオプションを、それぞれ与える必要があります。

この **data** オプションの形式は、次のとおりです。

principal_name key_type {*version*} {*key_value*}

ここで、

principal_name
これは、keytab ファイルが作成されるサーバー・プリンシパルの名前です。

key_type
これは、キーがプレーン・テキストあるいは DES 暗号化フォーマットで格納されるかどうかを指定するコードです。

- **des** は DES 暗号化を示します。
- **plain** はプレーン・テキストを指示します。

version これは、キーのバージョン番号です。バージョン番号を指定しないと、キーには番号 **1** が割り当てられます。

key_value

ランダムにキーを生成する **-random** オプションを指定しない場合、キーの値を与える必要があります。 *key_type* が **plain** の場合、キーをプレーン・テキストで提供します。 *key_type* が **des** の場合、 DES 暗号鍵を提供する必要があります。

以下のサンプル・コマンドは、次に示す作業を行います。

1. **dced** keytab オブジェクト **/.:/hosts/music/config/keytab/svr4_key** を作成する。
2. **music** という名前のローカル・マシン上の **keys** ディレクトリーに、**/opt/dcelocal/keys/svr4_key** という名前の keytab ファイルを作成する。
3. プリンシパル **mahler** のファイルにプレーン・テキストのキー・エントリーを作成し、これにバージョン番号 **3** を割り当てる。

```
dcecp> keytab create
/.:/hosts/music/config/keytab/svr4_key -attr ¥
> {{storage /opt/dcelocal/keys/svr4_key} ¥
> {data {mahler plain 3 mon#Repos}}}
dcecp>
```

Keytab ファイルへのエントリーの追加

既存の keytab ファイルにエントリーを追加するには、 **keytab add** コマンドを使用します。このコマンドを使用する場合、keytab ファイルの **dced** オブジェクトおよび、 409ページの表23 に記述されているオプションのいずれかを指定する必要があります。

次のコマンドは、サーバー・プリンシパル **svr_3** の **kfile_3** という名前の keytab ファイルにキーを追加します。キーは自動的に生成され、またレジストリーは keytab ファイルと同期するように更新されます。

```
dcecp> keytab add /.:/hosts/foo/config/keytab/kfile_3 ¥
> -member svr_3 -random -registry
dcecp>
```

keytab ファイルからのエントリーの削除

keytab ファイルのエントリーは、 **dcecp keytab remove** コマンドを使用して削除することができます。このコマンドを使用する場合は、 keytab ファイルの **dced** オブジェクトの名前を指定する必要があります。

keytab remove コマンドを使用する場合は、keytab ファイルの名前および、キーを削除するプリンシパル (あるいは一連のプリンシパル) の名前を指定する必要があります。

また、削除するキーのバージョン番号を指定する **-version** オプション、および削除するキーのタイプを指定する **-type** オプション (プレーン・テキスト・キーには **plain**、 DES 暗号鍵には **des**) を指定することができます。 **-version** あるいは **-type** オプションを使用すると、指定されたバージョンあるいはタイプのキーだけが削除されます。

次のコマンドは、keytab ファイル **/.:/hosts/foo/config/keytab/kfile_3** 内のプリンシパル **svr_2** のすべての DES キーを削除します。

```
dcecp> keytab remove /.:/hosts/foo/config/keytab/kfile_3 ¥
> -members svr_2 -type des
dcecp>
```

Keytab ファイルの削除

ローカル・ファイルと keytab ファイル、およびそれらに関連する **dced** オブジェクトは、 **dcecp keytab delete** コマンドを使用して削除することができます。

ローカル keytab ファイルと **dced** オブジェクトを削除するには、コマンドにローカル・ファイル名を指定します。名前を中括弧で囲み、またそれらをスペースで区切ることにより、 1 つのコマンドで複数の keytab ファイルを削除することができます。たとえば、次のコマンドは、keytab ファイルと **dced** オブジェクト **/.:/hosts/foo/config/keytab/kfile_2** および **/.:/hosts/foo/config/keytab/kfile_3** を削除します。

```
dcecp> keytab delete {/.:/hosts/foo/config/keytab/kfile_2 ¥
> /.:/hosts/foo/config/keytab/kfile_3}
dcecp>
```

dced オブジェクトだけを削除するには、**-entry** オプションを使用します。

たとえば、次のコマンドは **/.:/hosts/foo/config/keytab/kfile_3** という名前の **dced** オブジェクトを削除しますが、ローカル・ファイル **/opt/***dcelocal*/**keys/kfile_3** はそのままです。

```
dcecp> keytab delete -entry /.:/hosts/foo/config/keytab/kfile_3
dcecp>
```

Keytab ファイル内のサーバーとマシンのパスワードの変更

すべてのプリンシパルのパスワードは、期限切れになったら変更する必要があります。人間のプリンシパルは、自分のパスワードを変更するために、自分のプラットフォームの **chpass** コマンドを使用できます。 **dced** のセキュリティー妥当性検査サービスは、必要に応じて、ランダムに生成されたパスワードを割り当てることにより、マシンのパスワードを自動的に変更します。このデーモンは、DCE により提供され、またネットワーク・アクセスに関与するそれぞれのローカル・マシンで稼働します。一般に、他のベンダーが作成するサーバーあるいはアプリケーションも、必要に応じて、ランダムにパスワードを生成することによって、パスワードを自動的に変更すると想定することができます。ただし、サーバー自身の識別の下で稼働するサーバーが、自分のパスワードを自動的に更新しない場合は、ユーザーが **dcecp keytab add** コマンドを使用して手作業でこれを行う必要があります。これについては、411ページの『Keytab ファイルへのエントリーの追加』に説明されています。

注: 人間のプリンシパルの識別のもとで実行するサーバーは、自動的に自身のパスワードを更新してはなりません。そのようなサーバーが自身のパスワードを更新するとき、人間のプリンシパル (サーバーはそのプリンシパルの識別のもとで稼働している) のパスワードも更新します。次に、人間のプリンシパルは、システムにログインし、再度、認証するために、このランダムに生成されたパスワードを与えなければなりません。人間のプリンシパルはランダムに生成されたパスワードを知ることができないため、プリンシパルはシステムにログインできず、再認証ができません。

Keytab ファイル内のサーバーまたはマシンの汚されたパスワードの処理

サーバーあるいはマシンのパスワードが汚された場合、次のステップを行ってレジストリーとサーバーのローカル keytab ファイルのパスワードを変更する必要があります。

1. **keytab remove** コマンドを使用して、汚されたパスワードを削除します。
2. **keytab add** コマンドを使用して、サーバーまたはマシンの新しいパスワードを作成します。
3. **keytab add** コマンドの **registry** オプションを使用して、サーバーまたはマシンのレジストリー・アカウントを、サーバーまたはマシンの keytab ファイルと同時に更新しない場合は、 **account modify** コマンドを実行して、レジストリー内のサーバーまたはマシンのパスワードが keytab ファイルのパスワードと一致するよう変更してください。

ローカル・レジストリーの保守

ネットワーク・レジストリーが使用できない場合、ローカル・レジストリーを使用してそのマシンからログインすることができます。ローカル・レジストリーは、ネットワーク・レジストリー・サーバーが稼働している場合に、人間のユーザーまたは人間でないユーザーがローカル・マシンから DCE ログインを最初に実行したときに、自動的に作成されます。ユーザーがマシンにログインすると、ユーザーのアカウント情報がローカル・レジストリーに自動的に追加されます。

ローカル・レジストリー・データベースは、以下のファイルから構成されます。

dcelocal/**var/security/lrgy_data**
: アカウント情報エントリーが入っています。

dcelocal/**var/security/lrgy_tgts**
: チケット発行チケットのエントリーが入っています。

dcelocal/**var/security/lrgy_lock**
: レジストリーを読み取り / 書き込み操作からロックするために、セキュリティー・サーバーが使用します。

セキュリティー・コマンド **rgy_edit** を使用して、ローカル・レジストリーを保守する必要があります。以下のサブコマンドが使用可能です。

view ローカル・レジストリーのエントリーをすべて表示します。

delete *principal_name*
: *principal_name* で指定されたプリンシパルをレジストリーから削除します。

properties
: ローカル・レジストリーの属性を設定および表示します。

help ローカル・レジストリー上のヘルプ情報を獲得します。

上記のリストにあるサブコマンドに加えて、**rgy_edit** コマンドは、 1 つの追加サブコマンド **purge** および、*レジストリー容量* と呼ばれる追加のレジストリー属性を提供し、ローカル・レジストリー内のエントリーの数を制限します。

このセクションの残りの部分では、 **purge** サブコマンドとレジストリー容量属性について説明します。

レジストリー容量属性

rgy_edit properties コマンドは、ローカル・レジストリーが格納できるエントリーの数に制限を設けることができます。たとえば、レジストリー容量が **10** にセットされており、レジストリーに 10 個のエントリーがあるとします。新しいユーザーがローカル・マシンからログインすると、そのユーザーのエントリーが作成されます。この新しいエントリーは、元の 10 個のエントリーのうちの最も古いものを上書きします。ユーザーがマシンから初めてログインすると、新しく作成されたレジストリー・エントリーが最も古いエントリーを上書きします。

容量属性と存続期間属性のセット

ローカル・レジストリーが格納できるエントリー数に制限を設けることができます。この制限は、ローカル・レジストリー・オブジェクトの **Capacity** 属性で定義します。この属性の値を適度なエントリー数に設定すれば、ローカル・レジストリーを大きくなりすぎないように維持することができます。

たとえば、マシンのローカル・レジストリーの **Capacity** 属性が **10** にセットされており、そのローカル・レジストリーに 10 個のエントリーがあるとします。新しいユーザーがローカル・マシンからログインすると、そのユーザーのエントリーが作成されます。この新しいエントリーは、現在ローカル・レジストリーに存在する元の 10 個のエントリーのうち、最も古いものを上書きします。ユーザーがマシンから初めてログインすると、新しく作成されたレジストリー・エントリーが最も古いエントリーを上書きします。

ローカル・レジストリーの **Capacity** 属性をセットするには、 **rgy_edt properties** コマンドを使用します。

このコマンドを初めて入力すると、現在のレジストリー容量と存続期間が表示されることに注意してください。また、ローカル・レジストリーにあるアカウント・エントリーの数と、 TGT エントリーの数も表示されます。それぞれのアカウント・エントリーについて、アカウントの証明書を格納する TGT エントリーが 1 つあります。

```
$ rgy_edit -l
rgy_edit=> properties
Local Registry Properties:
  Capacity:   25 entries
  Contains:   6 account entries
  Contains:   6 tgt entries
  Lifespan:   3w
Do you wish to make changes [y/n]? (n) y
Enter local registry capacity: (25) 50
Enter acct lifespan in days or 'forever': (3w) 12w
```

上記の例で、**12w** は 12 週の存続期間を指定しています。存続期間を入力するとき、以下のものを入力することができます。

- **forever** - エントリーの存続期間が決して切れないことを指定します。ただし、レジストリー容量を超えるとエントリーは上書きされます。 **Enter acct lifespan in days or 'forever': (3w)** のプロンプトに対して **0** (ゼロ) を入力すると、 **forever** を入力したのと同じ結果になります。

- 10 進数 - 測定単位 (週、日、時間、分、秒) の数値を指定します。通常、この数値には、次に説明するような測定単位が伴います。測定単位を付けずに数値だけを入力すると、測定単位はデフォルトである「日」になります。
- **w**、**d**、**h**、**m**、または **s** - 以下の測定単位を指定します。
  - **w** 週
  - **d** 日
  - **h** 時間
  - **m** 分
  - **s** 秒

数値を付けた測定単位は、任意に組み合わせて使用することができます。たとえば、存続期間を 12 週間と 8 時間 30 秒に設定するには、次のコマンドを入力します。

```
Enter acct lifespan in days or 'forever': (3w) 12w8h30s
```

一連の数値と測定単位の最後を数値だけにすると、測定単位のない数値はデフォルトとして秒数になります。たとえば、次のように入力すると、存続期間は 12 週間と 30 秒であると想定されます。

```
Enter acct lifespan in days or 'forever': (3w) 12w30
```

有効期限が切れたエントリーの除去

rgy_edit purge サブコマンドは、有効期限が切れたエントリーをローカル・レジストリーから削除します。 **rgy_edt purge** コマンドを実行すると、コマンドは存続期間属性値の有効期限が切れているエントリーをすべて削除します。

上記の例では、ローカル・レジストリーの存続期間属性値を **12w** にリセットしています。存続期間を **12w** にセットしてあるので、次の **rgy_edt purge** コマンドは、 12 週よりも古いすべてのローカル・レジストリー・エントリー (アカウントと TGT) を削除します。

```
rgy_edit=> purge
1 account entries purged
1 TGT entries purged
```

第32章 拡張レジストリー属性の作成および使用

レジストリーは、プリンシパル、グループ、オーガニゼーションおよびアカウントについての特定の情報を格納しています。これは、**dcecp** コマンドを使用してプリンシパル、グループ、オーガニゼーションおよびアカウントを作成するときに作成する情報です。レジストリー・データベースに格納することができる情報の種類は、レジストリー・スキーマの中に定義されます。レジストリー・スキーマとは本来、データベースに格納されるデータの種類のカタログです。レジストリー・オブジェクトに関連付けるあるいは付加することができる属性の各タイプには、スキーマ・エントリー定義が存在します。たとえば、スキーマ・エントリーは、プリンシパル名を DCE PCS フォーマットの印刷可能文字ストリングとして定義します。プリンシパルを作成するとき、PCS 形式で格納されるテキスト・ストリングを入力します。

拡張レジストリー属性 (ERA) 機能を使用すれば、ユーザーの選択する属性型を定義する、スキーマ・エントリーを追加することができます。これらの属性は、レジストリー・スキーマを拡張するので、*拡張属性* と呼ばれます。拡張属性型が定義されたあと、 **dcecp create** あるいは **modify** 操作により、それらをセキュリティー・オブジェクトに付加することができます。ユーザーが作成する拡張属性型は、 DCE とともに実行されるカスタム・アプリケーションによって使用され、処理のためにこれらのアプリケーションに渡されます。たとえば、ユーザーの MVS™ 名を必要とする MVS アプリケーションを使用するとき、レジストリーに格納される MVS 名拡張属性を設定することができます。次に MVS 名は、MVS アプリケーションに渡されて適切に処理されます。

プリンシパルが拡張属性を持っている場合、これらの属性は、プリンシパルが認証されるときに入手される、特権属性証明書 (EPAC) で伝えられます。

本書では、属性型は、拡張属性型を定義するスキーマ・エントリーを指します。属性インスタンスは、レジストリー・オブジェクトに付加される属性を指し、値を持ちます。

ここでは、属性型と属性インスタンスの作成と保守を行う方法について説明します。最初に **xattrschema** オブジェクトについて説明し、次に属性型を定義し、属性をオブジェクトに付加する方法について説明します。

xattrschema オブジェクト

拡張属性型は、CDS ネームスペースのセキュリティー・ジャンクション・ポイント (通常、 **/.:/sec**) の **xattrschema** という名前のオブジェクトに格納されます。 **xattrschema** と、これに含まれる属性型定義へのアクセスは、 **xattrschema** オブジェクトの ACL によって制御されます。 **xattrschema** オブジェクトは、他のレジストリー・データと同様に、マスター・セキュリティー・サーバーからレプリカに伝搬されます。

属性型の作成および保守

属性型の作成と保守には、**dcecp xattrschema** コマンドを使用します。このコマンドの一部として、属性型の完全修飾名 (たとえば、 **/.:/sec/xattrschema/***name*) を引き数として指定する必要があります。

属性型の作成

属性型を作成するには、**dcecp xattrschema create** コマンドを使用します。このコマンドに指定できる属性オプションを以下に要約します。この表で説明されているオプションは、属性リストの中にダッシュを使用しないで指定できることに注意してください。

Options to Create Extended Attributes

オプション
意味

-aclmgr *description*
この属性型を付加できるオブジェクトをサポートする ACL マネージャーのタイプ、およびこれらのマネージャーがサポートする許可の必須のリスト。デフォルトはありません。属性型 ACL マネージャーについては、 421ページの『属性用の ACL マネージャーの定義』で詳細に説明されています。

-annotation *string*
属性型に注釈を付ける PCS テキスト・ストリング。ストリングにスペースが含まれる場合は、中括弧または引用符で囲みます。デフォルトはブランクです。

-applydefs
このオプションは現在、実装されていません。

-encoding *type*
属性型のインスタンス値の形式。属性のエンコードについては、 422ページの『属性型のエンコードの定義』に詳細に記述されています。

-intercell
このオプションは現在、実装されていません。

-multivalued {yes | no}
属性が複数値であるかどうかを示します (**yes** = 複数値、**no**= 複数値でない)。属性が複数値の場合、同じ属性型の複数インスタンスを単一のレジストリー・オブジェクトに付加することができます。たとえば、属性 A が複数値としてコード化されている場合、 1 つのプリンシパルが、属性 A の複数のインスタンスを持つことができます。複数値とコード化されていない場合は、プリンシパルは属性 A のインスタンスを 1 つだけを持つことができます。デフォルトは **no** です。

-reserved {yes | no}
属性が予約済みかどうかを示します (**yes** = 予約済み、**no**= 予約済みでない)。予約済みの属性型は、予約された制約が除去されるまで削除することはできません。デフォルトは **no** です。

-scope *name*
現行のリリースでは実装されていません。

-trigtype *type*

トリガー・サーバーが属性型に関連付けられるかどうか、およびトリガー・サーバーが関連付けられている場合はトリガーのタイプも示します。可能な値は、次のものです。 **none** - トリガー・サーバーはその属性型に関連付けられていない。 **query** - 照会トリガー・サーバーがその属性型に関連付けられている。 **update** - 更新トリガー・サーバーがその属性型に関連付けられている。 **-trigtype** オプションを **query** または **update** にセットした場合は、 **-trigbind** オプションを指定して、トリガー・サーバーのバインディングを指定する必要があります。詳細については、424ページの『属性トリガー・サーバーの定義』を参照してください。

-trigbind *binding*

トリガー・サーバーが属性型に関連付けられている場合、このオプションは、トリガー・サーバーのバインディングを指定します。

-unique {yes | no}

属性型の各インスタンスがセルの中で固有でなければならないかどうかを示します (**yes**= 固有、**no**= 固有ではない)。たとえば、属性型 A のインスタンスがセルの中の 25 のプリンシパルに付加されると仮定します。属性型 A を固有と指定すると、これら 25 のプリンシパルのそれぞれの A 属性の値は異なっていなければなりません。固有と指定されていない場合は、25 すべてのプリンシパルに、属性 A の同じ値を割り当てることができます。デフォルトは **no** です。

-uuid *uuid*

内部で属性型を識別する UUID。 **dcecp xattrschema create** コマンドへの引き数として指定された名前が、属性型のアクセスに使用されることに注意してください。 UUID が指定されていない場合は、システムがそれを生成します。

dcecp xattrschema create コマンドの構文は次のとおりです。

```
xattrschema create attr_name{attr_options}
```

ここで、

attr_name

これは、作成する属性型の完全修飾名です。

attr_option

これは、上記のリストに記述されている 1 つまたは複数のオプションです。

次のサンプル・コマンドは、**employee_num** の名前の拡張属性型を作成し、それに **principal** の ACL マネージャーと **integer** のエンコード・タイプを割り当てます。

```
dcecp> xattrschema create /.:/sec/xattrschema/employee_num ¥
> -aclmgr {principal r r r r} -encoding integer
dcecp>
```

上記の例では、情報を与えるためにオプションを使用していますが、標準の **dcecp** 属性リストを使用することもできます。

名前のリストを指定することにより、 1 回のオペレーションで複数のスキーマ・エントリーを作成することができます。ただし、1 つのコマンド引き数に複数のスキーマ名が含まれている場合は、 UUID 属性を指定することはできず、指定した属性は作成されるエントリーすべてに適用されることに注意してください。

属性型の変更

属性型を変更するには、**dcecp modify** コマンドと **-change** オプションを使用します。 **aclmgr**、**applydefs**、**intercell**、**trigbind**、 **annotation**、および **reserved** スキーマ・タイプ属性だけが変更できます。

xattrschema modify コマンドの構文は次のとおりです。

xattrschema modify *attr_name* **-change** *new_option*

ここで、

attr_name
これは、変更する属性型の完全修飾名です。

new_option
これは、変更を指定するオプションです。

次のサンプル・コマンドは、**MVSname** 属性の注釈を変更するために、この属性を変更します。コマンドには完全修飾属性型名を指定しなければならないことに注意してください。

```
dcecp> xattrschema modify
/.:/sec/xattrschema/MVSname -change ¥
> {annotation {Use with version 3.0}}
dcecp>
```

属性型の名前変更

拡張属性の名前を変更するには、**dcecp xattrschema rename** コマンドを使用します。コマンドは次の形式で入力します。

xattrschema rename *old_name* **-to** *new_name*

ここで、

old_name
これは、変更する完全修飾拡張属性名です。

new_name
これは、新しい拡張属性名です。

次の例は、**xattrschema rename** コマンドを使用して属性名を **log_name** から **MVSname** に変更します。

```
dcecp> xattrschema rename /.:/sec/xattrschema/log_name -to MVSname
dcecp>
```

属性型の削除

拡張属性を削除するには、**dcecp xattrschema delete** コマンドを使用します。属性型を削除すると、その属性型のすべてのインスタンスも削除されることに注意してください。たとえば、**MVSname** 属性のインスタンスが **delores** という名前の

プリンシパルに付加されているとします。 **MVSname** 属性を削除すると、**delores** に付加されているその属性のインスタンスも削除されます。

属性型を削除するには、次の形式でコマンドを入力します。

xattrschema delete *attribute_name*

ここで、*attribute_name* は削除する属性の完全修飾名です。

たとえば、**MVSname** という名前の拡張属性を削除するコマンドは次のようになります。

```
dcecp> xattrschema delete /.:/xattrschema/MVSname
dcecp>
```

属性用の ACL マネージャーの定義

拡張属性型を定義する場合、属性を付加できるオブジェクトとその属性をアクセスするための許可を定義する必要があります。このためには、属性型をいくつかの ACL マネージャーと関連付け、またこのタイプの属性インスタンスへのアクセスを制御する許可セットを指定します。属性を付加できる先は、 ACL マネージャー・セットに指定されている ACL マネージャー・タイプがサポートしているオブジェクトだけです。また、ACL マネージャー・セットに指定されている許可だけが、属性インスタンスへのアクセスに有効です。 (これらの許可は、 ACL マネージャーが制御するオブジェクトに対して ACL マネージャーがすでに設定している許可にさらに追加されることに注意してください。) たとえば、**MVSname** という名前の属性型の ACL マネージャー・セットが、プリンシパル用の ACL マネージャー・タイプだけをリストしているとします。その場合、**MVSname** という名前の属性型のインスタンスは、プリンシパルだけに付加することができ、他のレジストリー・オブジェクトには付加できません。 **MVSname** 属性用の ACL マネージャー・セットには、 **MVSname** 属性へのアクセスを制御する許可も含まれています。

属性の ACL マネージャー・セットを指定するには、 **dcecp xattrschema -aclmgr** オプションを使用します。このオプションの形式は次のとおりです。

{*mgr_uuid queryset updateset testset deleteset*}

ここで、

mgr_uuid
: これは、属性型に関連付ける ACL マネージャーを識別する UUID です。DCE が提供する ACL マネージャー・タイプにアクセスするために、UUID あるいは次の短縮名 (これらは内部で UUID に変換される) の 1 つを指定することができます。

  policy ポリシー・オブジェクト用の ACL マネージャーにアクセスします。

  principal
  : プリンシパル用の ACL マネージャーにアクセスします。

  group グループ用の ACL マネージャーにアクセスします。

  organization
  : オーガニゼーション用の ACL マネージャーにアクセスします。

secdirectory
レジストリー・データベースのディレクトリー用の ACL マネージャーにアクセスします。

replist レプリカ・リスト用の ACL マネージャーにアクセスします。

xattrschema
レジストリー・スキーマ用の ACL マネージャーにアクセスします。

srvrconf
dced オブジェクト用の ACL マネージャーにアクセスします。

queryset
これは、属性のインスタンスを照会する許可セットです。

updateset
これは、属性のインスタンスを変更する許可セットです。

testset これは、属性のインスタンスをテストする許可セットです。

deleteset
これは、属性のインスタンスを削除する許可セットです。

複数の許可を持つ許可セットを入力するには、許可を連結します。たとえば、許可 **t**、**M**、および **d** を入力するには、**tMd** と入力します。

各 ACL マネージャーの情報は中括弧で囲み、各項目の間にスペースを入れます (ただし、連結した許可セットの項目の間は除く)。

たとえば、**MVS_name** 属性のための追加の ACL マネージャーを定義する、次のコマンドを考えてみます。

```
dcecp> xattrschema
modify /.:/sec/xattrschema/MVS_name ¥
> -aclmgr {18dbdad2-23df-11cd-82d4-080009251352 r w t mD}
dcecp>
```

このコマンドは、UUID **18dbdad2-23df-11cd-82d4-08000925135** によって識別される、 ACL マネージャーを **MVS_name** 属性に追加します。 ACL マネージャーの許可セットは、次のとおりです。

- **r** は照会許可セット
- **w** は更新許可セット
- **t** はテスト許可セット
- **mD** は削除許可セット

属性型の ACL マネージャー・セットは変更あるいは削除できないことに注意してください。ただし、それに追加のマネージャー・タイプを追加することはできます。

属性型のエンコードの定義

属性型のエンコードで属性型に指定することができる、値のフォーマットを定義する必要があります。属性には、エンコードで定義されているフォーマットの値だけ

が割り当てられます。たとえば、エンコードが、この属性型のインスタンスには UUID 形式の値だけが含まれていることを指定することができます。

各属性型はエンコードを 1 つだけ持つことができ、このエンコードは変更することができません。その他に、特別のエンコード・タイプにより属性セットを作成することができます。

属性のエンコードを指定するには、**dcecp xattrschema -encoding** オプションを使用します。このオプションの形式は次のとおりです。

-encoding *type*

type パラメーターは、以下に記述されているエンコード・タイプのうちの 1 つです。

Encoding Types

エンコード・タイプ
: **意味**

any
: 属性値はすべてのエンコードを受け入れます。このエンコード・タイプは、スキーマ・エントリー内の ERA の定義に対してのみ有効です。 ERA のすべてのインスタンスは、他の値のエンコードを持っていなければなりません。

attrset
: 属性値は、中括弧で囲まれた属性型 UUID のリストでなければなりません。このエンコード・タイプは属性セットを定義します。属性セットを使用すると、属性の検索と探索を簡単に行えます。たとえば、属性セットの照会は、そのセットのメンバーである、属性のすべてのインスタンスを返します。

binding
: 属性値は、DCE サーバーとのコミュニケーションに適した、認証、許可、およびバインディング情報で構成されていなければなりません。オブジェクト属性として名前あるいはサーバー・バインディングを格納する場合には、このエンコードを使用してください。詳細については、426ページの『バインディング情報の指定』を参照してください。

byte
: 属性値は、バイトの列でなければなりません。バイト列はピックルと見なされるか、そうでなければ、自己記述型です。このエンコード・タイプでは、2 進データの入力が可能であることに注意してください。バイナリー・データの入力の詳細については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

confidential
: DCE のこのリリースには実装されていません。

i18ndata
: 属性値は、データをエンコード化するために使用される OSF レジストリー・コード・セットを識別するタグを持つ、国際化されたバイト列でなければなりません。このエンコード・タイプでは、2 進データの入力が可能であることに注意してください。バイナリー・データの入力方法の詳細については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

integer
属性値は、符号付き 32 ビットの整数でなければなりません。

printstring
属性インスタンス値は、PCS により印刷可能な文字ストリングでなければなりません。

stringarray
属性値は、1 つまたは複数の printstring の配列でなければなりません。
printstring はヌルでもよいことに注意してください。
属性値は、DCE UUID でなければなりません。

void 属性は値を持ちません。これは存在するか、あるいは存在しない、単なるマーカーです。

属性トリガー・サーバーの定義

いくつかの属性型は、入力属性値を検査するため、あるいはこれらの値が外部データベースに格納されるときに出力属性値を与えるために、外部サーバーのサポートを必要とします。このようなサーバーは、たとえば、従来のレジストリー・システムを DCE レジストリーに接続できます。属性トリガー機能は、*属性トリガー* と呼ばれる、外部 DCE サーバーへの自動呼び出しを提供します。

トリガー・サーバーは、トリガー・サーバーに関連付けられた属性が照会あるいは変更されると、自動的に呼び出されます。トリガー・サーバーが保守する情報へのアクセスは、そのサーバーによって完全に制御されることに注意してください。

注: 更新トリガー・サーバーは、本リリースではサポートされていません。

属性型をトリガー・サーバーに関連付けるには、 **-trigtype** および **-trigbind** **dcecp xattrschema** オプションを使用します。

-trigtype オプション

-trigtype オプションは、属性型がトリガー・サーバーに関連付けられるかどうか、そして、関連付けられる場合は、サーバーの種類を定義します。このオプションの形式は次のとおりです。

```
-trigtype [none | query | update]
```

ここで、

none これは、属性がトリガー・サーバーに関連付けられないことを示します。(これは、デフォルトです。)

query これは、属性が照会トリガーに関連付けられることを示します。照会トリガー・サーバーは、照会だけを行うことができます。

update
これは、属性が更新トリガーに関連付けられることを示します。更新トリガー・サーバーは、照会と更新を行うことができます。

注: 更新トリガー・サーバーは、本リリースではサポートされていません。

一度セットした **-trigtype** オプションは変更できません。

-trigbind オプション

-trigbind オプションは、トリガー・サーバーおよびトリガー・バインディング自体の認証情報を定義します。

-trigbind オプションの形式は次のとおりです。

-trigbind {{*auth_info*}{*binding_info*}}

次のセクションでは、認証タイプとバインディングを指定する方法について説明します。

認証タイプの指定

auth_info パラメーターの構文は次のとおりです。

{*auth_serv_type name prot_level authentication_service authorization_service*}

ここで、

auth_serv_type
次に示す認証タイプのいずれかを指定します。

- **none** - 認証は行われません。
- **dce** - 標準の DCE 認証が行われます。

認証を使用しない場合、バインディング自体を除き、他の情報は必要ありません。標準の DCE 認証タイプを使用する場合、残りのすべてのパラメーターを指定する必要があります。

name トリガー・サーバーのプリンシパル名を指定します。

prot_level
クライアントとサーバーの間の認証されたコミュニケーションが認証サービスによって保護される程度を決める、保護レベルを指定します。指定できる保護レベルは以下のものです。

- **default** - デフォルトの保護レベルである **pkt** を使用します。
- **none** - 認証を行いません。チケットは交換されず、セッション・キーは設定されず、クライアント EPAC あるいは名前は証明されず、伝送は暗号化されずに行われます。証明されていない EPAC は信頼すべきではありませんが、それらは、デバッグ、トレースおよび測定目的には使用できます。
- **connect** - クライアントがサーバーとの関係を確立する際にだけ認証を行います。
- **call** - サーバーが要求を受け取ったとき、それぞれのリモート・プロシージャー・コールの始めにだけ認証を行います。

  このレベルは、接続ベースのプロトコル・シーケンス (つまり、**ncacn_ip_tcp**) によるリモート・プロシージャー・コールには適用されません。このレベルが指定され、しかもバインディング・ハンドルが接続ベースのプロトコル・シーケンスを使用する場合、ルーチンは、**pkt** 保護レベルを代わりに使用します。
- **pkt** - 受け取るデータがすべて、予期したクライアントからのものであることを保証します。

- **pktinteg** - クライアントとサーバーの間で伝送されるすべてのデータが変更されていないことを保証し、検証します。これは、RPC ランタイムに提供されることが保証されている、最高の保護レベルです。
- **pktprivacy** - これまでのすべてのレベルで指定された認証を行い、さらにそれぞれの RPC 引き数値を暗号化します。これは最高の保護レベルですが、 RPC ランタイムに提供されることは保証されていません。

authentication_service
認証サービスを指定します。認証サービスで提供される正確な保護レベルは、保護レベルによって指定されます。次の認証サービスがサポートされます。

- **default** - DCE 機密共用キー。
- **none** - 認証なし。チケットは交換されず、セッション・キーは設定されず、クライアント EPAC あるいは名前は伝送されず、また伝送は暗号化されずに行われます。このバインディングを使用して行われたリモート・プロシージャー・コールの認証をオフにするには、 **none** を指定します。
- **secret** - DCE 機密共用キー認証。

authorization_service
許可サービスを指定します。アプリケーション・データなどの認可データの有効性と信頼性は、認証サービスおよび指定されている保護レベルによって決まります。次の認可サービスがサポートされます。

- **none** - サーバーは認可を行いません。これは、認証の行われないことを示す **none** に認可サービスがセットされている場合にのみ有効です。
- **name** - サーバーは、クライアントのプリンシパル名に基づいて認可を行います。この値は、認可サービスが **none** の場合には使用できません。
- **dce** - サーバーは、このバインディングを使用して行われる各リモート・プロシージャー・コールによりサーバーに送られる、クライアントの DCE EPAC を使用して認可を行います。通常、アクセスは DCE ACL に照らして検査されます。

バインディング情報の指定

binding_info パラメーターは、バインディングを指定します。バインディングとしては、ストリング・バインディング、サーバー・エントリー名、あるいはいくつかのストリング・バインディングあるいはサーバー・エントリー名が含まれているリストがあります。次の例は、サーバー・エントリー名バインディングを示しています。

```
./.:/hosts/host_name/dce_entity_name
```

次の例は、標準構文のストリング・バインディングを示しています。

```
ncadg_udp_ip:130.105.96.3[1234]
```

次の例は、TCL 構文のストリング・バインディングを示しています。

```
ncarn_ip_tcp
10-29.58.00 2001
```

-trigbind オプションのサンプル値

次の例は、**-trigbind** オプションの値を示しています。この例では、バインディングは **MVS_server** のプリンシパル名を持ち、パケット・プライバシー保護レベルで認証され、また機密共用キーと DCE の認可サービスを使用します。バインディングは、サーバー・エントリー名として提供されます。

```
-trigbind {{dce MVS_server pktprivacy secret dce} ¥
    {/.:/hosts/host_name/dce_entity_name}}
```

属性インスタンスの作成および保守

dcecp を使用すれば、オブジェクトへの拡張レジストリー属性の付加、それらの属性に割り当てられた値の変更、および、オブジェクトに付加された他の属性に対するのと同じように、アタッチメントの削除を行うことができます。

dcecp create および **modify** 操作を使用して、次に示すレジストリー・オブジェクトに拡張レジストリー属性を付加することができます。

- **principal**
- **group**
- **organization**
- **policy**

  注: DCE バージョン 1.2.1 では、ポリシー (policy) ・オブジェクトに属性を付加することはできません。
- **directory**
- **replist**
- **xattrschema**

オブジェクトへの属性インスタンスの付加

dcecp principal -attribute 操作を使用してオブジェクトを作成するとき、オブジェクトに属性を付加することができます。また、 **dcecp modify -add** 操作を使用して、既存のオブジェクトに属性を付加することができます。

たとえば、プリンシパル **delores** を作成し、同時に、**admin** の値をもつ **MVSname** 属性を付加するには、次の **principal create** コマンドを使用します。

```
dcecp> principal create delores -attribute {MVSname admin}
dcecp>
```

admin の値をもつ **MVSname** 属性を **delores** という名前のプリンシパルに付加するには、次の **principal modify** コマンドを使用します。

```
dcecp> principal modify delores -add {MVSname admin}
dcecp>
```

複数値拡張属性のインスタンスを追加するには、属性名の後に、各値をスペースで区切って入れます。たとえば、**value1**、**value2**、**value3**、および **value4** の値をもつ **multi_name** 属性を、 **delores** という名前のプリンシパルに付加するには、次のコマンドを使用します。

```
dcecp> principal modify delores -add {multi_name value1 value2 ¥
> value3 value4}
dcecp>
```

属性インスタンスの変更

属性インスタンスの値を変更するには、**dcecp modify -change** 操作を使用します。属性が変更可能かどうかは、その属性を使用するアプリケーションによって判別されます。たとえば、次のコマンドは、プリンシパル **delores** の **MVSname** に割り当てられている値を、 **admin** から **cell_admin** に変更します。

```
dcecp> principal modify delores -change {MVSname cell_admin}
dcecp>
```

複数値属性の値を変更するために上記の **dcecp modify -change** コマンドを使用すると、複数値属性のすべてのインスタンスが削除され、コマンドに指定された新しい値で置き換えられます。たとえば、特定の値だけを変更する場合は、すべての値を入力する必要があります。たとえば、**multi_name** 属性が **value1**、**value2**、**value3**、および **value4** の 4 つの値を持っているとします。 **value4** を **value5** に変更するには、次のコマンドを入力しなければなりません。

```
dcecp> principal
modify delores -change {multi_name {value1 value2 ¥
> value3 value5}}
dcecp>
```

ただし、複数値属性に個別の値を追加あるいは削除することができます。値を追加するには、**-add** オプションを使用します。たとえば、**multi_name** 属性が、値 **value1**、**value2**、 **value3**、および **value5** を持っているとします。次のサンプル・コマンドは、**value6** を **multi_name** 属性に追加します。

```
dcecp> principal modify
delores -add {multi_name value6}
dcecp>
```

(複数値属性の特定の値を削除するには、次に説明する **remove** オプションを使用してください。)

次のコマンドは、**delores** という名前のプリンシパルに付加されている **multi_name** という名前の属性のすべてのインスタンスを、**value1** の値を持つ 1 つのインスタンスで置き換えることに注意してください。

```
dcecp> principal modify delores -change {multi_name value1}
dcecp>
```

たとえば、**multi_name** 属性が次の値を持っているとします。

```
{multi_name value1 value2 value3}
```

この場合、上記のコマンドは値を次のように変更します。

```
{multi_name value1}
```

属性インスタンスの削除

オブジェクトに付加された属性インスタンスを削除するには、 **-remove** オプションを指定した **dcecp modify** コマンドを使用します。オブジェクトから属性のすべてのインスタンスを削除するには、 **-remove** オプションに属性名を指定します。たとえば、次のコマンドは、 **delores** という名前のプリンシパルから **MVSname** 属性のすべてのインスタンスを削除します。

```
dcecp> principal modify delores -remove MVSname
dcecp>
```

複数値属性から 1 つのインスタンスを取り除くには、属性名と属性値を指定します。たとえば、次のコマンドは **multi-value** という名前の複数値属性から、インスタンス **value5** だけを削除します。他の値と属性自体は、そのまま変わりません。

```
dcecp> principal modify delores -remove {multi-value value5}
dcecp>
```

ただし、複数値属性の最後のインスタンスを削除すると、値を持たない属性はオブジェクトに付加できないので、 **dcecp** はオブジェクトから属性を削除します。複数値属性の削除と、複数値属性からの値の削除を、同じコマンドで組み合わせることはできないので、注意してください。

オブジェクトから複数の属性を削除するには、 **-types** オプションを使用する必要があります。このオプションは、指定された値はすべて属性型の名前であり、属性値ではないことを **dcecp** に伝えます。たとえば、次のサンプル・コマンドは、**-types** オプションを使用して、プリンシパル **delores** から **MVSname** と **MVSinteger** という名前の属性を削除します。

```
dcecp> principal modify delores -remove {MVSname MVSinteger}
-types
dcecp>
```

-types オプションがないと、 **dcecp** は **MVSinteger** が **MVSname** 属性の値であると見なし、そのような値が存在しないので、コマンドは失敗します。

属性セットの使用

属性セットは、セットのメンバーである属性インスタンスを識別する、属性 UUID の集合です。属性セットを使用すると、オブジェクト上の関連する属性インスタンスをグループ化して、アクセスを容易にすることができます。たとえば、**dcecp show** 操作を使用して属性セットを表示する場合、表示は属性セットを拡張して、そのセットのすべてのメンバーを表示出力に含めます。この属性の拡張は、情報を表示する **dcecp** コマンドについてだけ働きます。属性インスタンスを作成および変更するコマンドは、コマンドに指定されている特定の属性に対してだけ作用します。セット・メンバーである属性は、属性セットとは独立して存在するため、他の属性と同様に、直接処理することができます。

各属性セットがオブジェクトに付加されます。また、システムは強要しませんが、セットのメンバーである各属性は、同一のオブジェクトに付加すべきです。属性セットをネストすることはできません。つまり、属性セットのメンバー自身を属性セットにすることはできません。

属性セットのメンバーを作成、変更、および削除するには、複数値属性の作成、変更、および削除のための指示に従ってください。セットのメンバーである属性インスタンスは、 UUID によって識別されます。

拡張レジストリー属性の制約事項

拡張レジストリー属性 (ERA) のスキーマを作成する場合、 **-unique** 属性 (**sec_attr_sch_entry_unique**) のみを推奨します。この属性が ERA スキーマに対して **yes** に設定されたとしても、その動作は **no** に設定されているかのようになります。 DCE コードは、オブジェクトに添付されている ERA 値の固有性を検査したり、強制実行したりしません。

ERA スキーマを作成する場合、 **-applydefs** 属性 (**sec_attr_sch_entry_use_defaults**) のみを推奨します。この属性が ERA スキーマに対して **yes** に設定されたとしても、その動作は **no** に設定されているかのようになります。 ERA がオブジェクトに明示的に添付されていない場合、 DCE コードは ERA に対してデフォルト値を提供しません。

ERA スキーマを作成する場合、 **-intercell** 属性のみを推奨します。この属性が **accept** (**sec_attr_intercell_act_accept**) または **evaluate** (**sec_attr_intercell_act_evaluate**) に設定されたとしても、その動作は **reject** (**sec_attr_intercell_act_reject**) に設定されているかのようになります。プリンシパルの EPAC がセル間のアクセスに使用される場合、 DCE コードは、プリンシパル、グループ、またはオーガニゼーションのすべての ERA 値を破棄します。

ERA スキーマを作成する場合、 **-encoding** 属性の **confidential_bytes** 値は、実装されません。 ERA スキーマの **-encoding** 属性を **confidential_bytes** (**sec_attr_enc_confidential_bytes**) に設定すると、 **bytes** (**sec_attr_enc_bytes**) と同じ動作をします。

ERA スキーマをエンコードしている属性セットの処理は、完全にはサポートされていません。 ERA スキーマ内でエンコードしている **attr_set** (**sec_attr_enc_attr_set**) を設定すると、 1 つの固有の uuid の下にある一連のスキーマ uuid をグループ化することができるようになります。 **sec_rgy_attr_lookup_by_id( )** API は、現在 **sec_rgy_lookup_no_expand( )** のように動作します。そのコンポーネントの中に設定されている属性のシームレス拡張は、現在サポートされていません。

ERA スキーマを作成する場合、 **-trigtype** 属性の **update** 値 (**sec_attr_trig_type_update**) は、実装されません。 ERA スキーマの更新トリガーは、 ERA 値が更新される前に登録済みのサーバーを検査する機能を提供しますが、この機能はまだ実装されていません。

第33章 マルチセル環境の管理

本書のこれまでの章では、個々のセルで行われる DCE 管理作業について説明しました。外部セルのプリンシパルがローカル・セルのオブジェクトにアクセスするマルチセル環境の管理では、異なるセル間でのプリンシパルの対話に起因する、追加の作業と考慮事項があります。

実際、2 種類のシステム管理者を持つことができます。 1 つはローカル・セルの管理用で、もう 1 つはマルチセル環境でのセル間の管理用です。 2 種類の管理者に対してグループを設定する場合、レジストリー・データベース内に、セル・プリンシパルを含む **krbtgt** ディレクトリー用の ACL をセットし、セル間管理者のグループだけが更新できるようにします。ただし、必ずその他のすべてのユーザーが **krbtgt** ディレクトリーへの読み取りアクセスができるようにします。さもないと、それらのユーザーのセル間アクセスは拒否されます。このように **krbtgt** ディレクトリーを保護する場合は、 **krbtgt** ディレクトリーの下のすべてのディレクトリーも適切な ACL を持っていることを確認してください。これを行う最も簡単な方法は、レジストリーが作成された後、 **krbtgt** ディレクトリー上の ACLs (オブジェクト ACL と初期作成 ACL) を変更することです。

この章では、外部セルのプリンシパルがローカル・セルのオブジェクトへアクセス、およびその逆のアクセスを可能にする、セル間の信頼関係について説明します。

信頼関係

注: *OSF DCE バージョン 1.2.2 のコードは、このセクションで説明する推移的信頼関係をサポートしていません。*

他のセルのプリンシパルに対して、ローカルのセルのオブジェクトへの認証アクセスを明示的に許可するためには、そのセルと信頼関係を確立する必要があります。これを行うには、**dcecp registry connect** コマンドを使用して、 2 つの特殊なアカウントを作成します。 1 つはローカル・セルのレジストリー内の、外部セルを表すアカウントであり、もう 1 つは外部セルのレジストリー内の、ローカル・セルを表すアカウントです。これらのアカウントを確立すると、それは外部セルの認証サービスが外部ユーザーを正しく認証することを信頼したことを意味し、したがって、このセルからのユーザーが、外部セルの認証サービスにより認証されたというマークを付けている場合、そのようなユーザーはすべて認証されているものと見なすことを示します。

信頼関係が確立されると、ローカル・セルのプリンシパルに対して行うのと同様に、 ACL エントリーを用いて、特定のオブジェクトへの外部プリンシパルのアクセスを制御することができます。また信頼関係があれば、外部セルのユーザーはローカル・セルのアカウントにログインすることもでき、その逆 (ローカル・セルのユーザーが外部セルのアカウントへログインすること) も可能になります。

他のセルのプリンシパルが自分のセルのオブジェクトに認証アクセスを行えるようにする信頼関係には 2 種類があります。この 2 種類の関係は、*直接信頼関係* と*階*

層推移的信頼関係 です。この章では、*推移的信頼関係* という用語は、階層推移的信頼関係の DCE の実装を表すために使用します。

直接信頼関係

直接信頼関係では、2 つのセルの認証サービスは認証キーを共用し、また互いに信頼して、それぞれのセルのプリンシパルを認証します。したがって、各セルからのユーザーが、そのセルの認証サービスにより認証済みのマークが付けられている場合、両方のセルはそれらのすべてのユーザーを認証されるべきものと見なします。共用認証キーは、認証されようとする 1 つのセルからもう 1 つのセルまでの、すべてのプリンシパルが使用する、単一のパスワード (セルごとに 1 つ) から取り出されます。直接信頼関係に関係するセルは、2 つだけです。

信頼関係の確立

直接信頼関係および、推移的信頼関係を確立するには、 **registry connect** コマンドを使用します。ただし、直接信頼関係は任意の 2 つのセルの間で作成できますが、推移的信頼関係は推移的承認パスによって結ばれたセルのみに対して作成できることに注意してください。

このコマンドは、次の 2 つの特別のアカウントを作成します。 1 つは、ローカル・セルのレジストリー内で外部セルを表すものであり、他の 1 つは、外部セルのレジストリー内でローカル・セルを表すものです。このコマンドは、同時にアカウントのプリンシパルを作成します。信頼関係が確立されれば、外部セルのユーザーはローカル・セルのアカウントにログインでき、その逆も可能です。特定のオブジェクトへの外部プリンシパルのアクセスは、ローカル・セルのプリンシパルに対して行うのと同様に、ACL エントリーにより制御します。

アカウント作成時に、 **registry connect** コマンドは 2 つのタスクを行うので注意する必要があります。最初のタスクは、両方のアカウントが共用する *1* つ のパスワードを自動的に生成することです。これは、自身のセルと信頼関係を持つセルにログインするユーザーは、同じプリンシパルと見なされ、また同じパスワードを共用することを意味します。 2 番目のタスクとして、**registry modify** コマンドは、所定の外部セル内のすべてのプリンシパルが共有する 1 つの UNIX 番号を生成します。この共用 UNIX 番号は、ローカル・マシン上のオブジェクトがアクセスされるとき、ローカル・プリンシパルと外部プリンシパルの UNIX 番号の間の衝突を回避するのに役立ちます。

レジストリーの内部で、またネットワーク・アクセスのために、プリンシパルはその完全修飾名を表す UUID によって識別されます。たとえば、プリンシパル **mahler** の場合、 UUID は **/.../dresden.com/dce/users/mahler** です。ただし、ローカル・マシン上のローカル・オペレーティング・システムは、 UNIX 番号によってプリンシパルを識別します。 UNIX 番号は、セルの間で固有である必要はないため、異なるセルの 2 つのプリンシパルが同じ UNIX 番号を持つことがあります。したがって、ローカル・セルのファイルをアクセスする外部プリンシパルが、ローカル・プリンシパルと同じ UNIX 番号を持つことがあり、ローカル・システムによって、ローカル・マシン上のローカル・ユーザーのファイルの所有者であると見なされます。

ローカル・セルにアクセスする特定のセルのすべてのプリンシパルに適用される UNIX 番号を作成すれば、このようなことが起きることを避けることができます。ただし、すべての外部ユーザーが同じ UNIX 番号を持つため、ローカル・ユーザーのファイルへのアクセスを防止する機能そのものが、それぞれの相手のファイルへのアクセスを可能にすることを知っておく必要があります。同じ外部セルからの各ユーザーは同一ユーザーと見なされるため、外部ユーザーが所有するローカル・マシン上の各ファイルは、同じ外部セルからの他のすべての外部ユーザーがアクセスできることになります。

信頼関係の作成

対等関係を作成するには、次のステップに従ってください。

1. **registry connect** コマンドを実行して、セル間認証アカウント (ローカル・セルのレジストリー内のアカウントと外部セルのレジストリー内のアカウント) を作成します。

   注: **rgy_edit cell** または **dcecp registry connect** コマンドが出されますが、セルの一方が (直前の信頼構成から) そのレジストリー内の他方のセル用の既存の **krbtgt** エントリーを持っている場合、コマンドは一見成功するかのように見えます。しかし、2 つの **krbtgt** エントリーのキーが同期していないため、認証されたセル間アクセスは失敗します。

   この状態は、セル間用に作成された 2 つの **krbtgt** アカウントのそれぞれをフルスクリーン表示にすることによって検出できます。アカウントは、**krbtgt/***foreign name* と命名され、一方のアカウントは、他方のセルにセル間アクセスできるようにするために各セルのレジストリー内に作成されます。外部セル用のこの **krbtgt** アカウントの作成時間が最新の変更時間と異なる場合、このエントリーが無効である場合もあります。

   この状態から回復するには、外部セル用の **krbtgt** アカウントおよびプリンシパルを削除します。これは、各セル内で行う必要があります。次に、**rgy_edit cell** または **dcecp registry connect** コマンドを再発行してください。 **krbtgt** アカウントおよびプリンシパルは、以下の **dcecp** コマンドを使用して削除できます。

   ```
   dcecp -c account delete krbtgt/
   dcecp -c principal delete krbtgt/
   ```

2. オプションとして、**account modify** コマンドを使用して、アカウント作成時にデフォルトとして割り当てられたアカウントの属性を微調整します。たとえば、アカウントの有効期限 (**expdate** 属性) は、デフォルトが **none** になっています。アカウントが、一定期間後に必ず更新されるようにするため、日付を入力したい場合があります。
3. アカウントが有効であることを示すため、外部セルのシステム管理者がローカル・セルを表すアカウントの **acctvalid** フラグを **yes** に変更したことを確認してください。片方または両方のアカウントが無効の場合、セル間のコミュニケーションは行われません。

registry connect コマンドのコマンド・オプション

registry connect コマンドを使用する場合、対等関係を確立する外部セルの完全修飾名を指定する必要があります。この名前は全パス名から取り出され、**krbtgt** という接頭部が付けられ、そして、アカウントのプリンシパルの 1 次名として使用されます。たとえば、セル名 **/.../dresden.com** を入力すると、プリンシパル名は **krbtgt/dresden.com** となります。未変更のセル名は、プリンシパルのフルネームとして格納されます。

registry connect は、ローカル・セル名をローカル・セルのアカウント・プリンシパルの 1 次名として使用することに注意してください。この名前は、全パス名から取り出されて、外部セル名と同様に、 **krbtgt** という接頭部が付けられます。

registry connect コマンドには次のオプションを指定することができます。

-acctvalid, -facctvalid
アカウントが有効であることをマークする設定。有効なローカル・アカウント (**-acctvalid**) は、外部セルからのユーザーがローカル・セル内のノードにログインできるようにします。有効な外部アカウント (-facctvalid) は、ローカル・セルからのユーザーが外部セル内のノードにログインできるようにします。各オプションのデフォルトは **invalid** です。

-expdate
ローカル・セルと外部セルのアカウントの有効期限が切れ、対等関係が終了し、 2 つのセルのプリンシパルの間での認証された通信がそれ以上できなくなる時刻および日付。アカウントを更新するには、このフィールドの日付を変更します。デフォルトは **none** です。

-facct, -facctpw
外部セルのシステム管理者は、外部セルのアカウントの名前とパスワードをユーザーに提供する必要があります。外部アカウントは、プリンシパルとアカウントの作成に必要な許可を持っていなければなりません。ユーザーのセルを表すアカウントを外部アカウントのレジストリーに作成するためには、ユーザーは外部レジストリーにアクセスするアカウントを持っていなければなりません。このアカウントの存続時間と作成割り当て量は、タスクを完了するために必要なものに制限する必要があります。

-group, -fgroup
ローカル・セル (**-group**) および外部セル (**-fgroup**) のアカウントに関連付けられるグループの名前。これらのグループは、アカウントには意味を持たず、また、外部セルあるいはローカル・セル内のいずれのユーザーとも関連しません。これは、すべてのアカウントはグループに関連させなければならない、というレジストリーの要件に従うために入力しなければなりません。そのグループが存在しなければ、それが作成されます。

-mypwd
registry connect コマンドは、作成されているアカウントのパスワードの入力を要求しません。このパスワードはランダムに生成されます。しかし、このコマンドの入力者は、自分の識別を検査させるために、自分のパスワードを **-mypw** オプションとともに指定する必要があります。

-org, -forg
ローカル・セル (**-org**) および外部セル (**-forg**) のアカウントに関連付けら

れるオーガニゼーション名。これらのオーガニゼーションは、アカウントには意味を持ちません、また、外部セルあるいはローカル・セル内のいずれのユーザーとも関連しません。これは、すべてのアカウントはオーガニゼーションに関連付けられる必要がある、というレジストリーの要件に従うために入力しなければなりません。そのオーガニゼーションが存在しなければ、それが作成されます。

セル間認証アカウントの作成例

次の例は、**registry connect** コマンドを使用して、 **/.../dresden.com** で識別される外部セルのためのアカウントを作成しています。ローカル・アカウントは、グループ **cell_group_local**、オーガニゼーション **cell_group_dres**、およびオーガニゼーション **cell_org_dres** に関連付けられます。アカウントの有効期限は、デフォルトの **none** です。

```
dcecp> registry connect /.../dresden.com -facct cell_log
-facctpw music ¥
> -group cell_group_local -fgroup cell_group_dres ¥
> -org cell_org_local -forg cell_org_dres -mypwd cell_admin
dcecp>
```

registry connect コマンドにより作成されるアカウント

registry connect コマンドにより作成されるアカウントとプリンシパルには、 表 24 にリストされているデフォルトの属性値が与えられます。これらの属性は、すべての外部プリンシパルがセルのオブジェクトをアクセスするときに、それらに適用されます。同様に、外部セル内にローカル・セルのために作成されるアカウントの属性は、ローカル・セルのプリンシパルが外部セルのオブジェクトにアクセスするときに、ローカル・セルのすべてのプリンシパルに適用されます。

表 24. セル間許可プリンシパルおよびアカウントのデフォルト属性値

情報	意味
アカウントのプリンシパル名	ローカル・セルのアカウントの場合はローカル・セル名、あるいは外部セルのアカウントの場合は外部セル名であり、全パス名から取り出され、 **krbtgt** という接頭部が付けられます。
fullname	セルのパス名。
quota	**none** にセットされます。この割り当て量は、セル間認証アカウントを使用して外部セルのオブジェクトにアクセスする、すべてのプリンシパルに適用されます。たとえば、オブジェクト作成割り当て量を 10 に変更すると、セルへのアクセスのためにそのアカウントを使用する外部ユーザーは、合計で 10 を超えるオブジェクトをセルのレジストリーに作成することができません。これは、外部プリンシパルごとに 10 ではありません。外部セル内のユーザー・セルのアカウントにセットされるオブジェクト作成割り当て量は、ユーザー・セルのプリンシパルが外部セルのレジストリーに作成することができるオブジェクトの数も同様に制限します。

表 24. セル間許可プリンシパルおよびアカウントのデフォルト属性値 (続き)

情報	意味
description、**home**、**shell**	ブランクにセットされます。
server	**yes** にセットされます。つまり、アカウントは、認証通信を行うことができるサーバーです。
client	**no** にセットされます。
pwdvalid	**yes** (有効) にセットされます。
acctvalid	**-acctvalid** および **-facctvalid** オプションが使用されていない場合、 **no** (無効) にセットされます。
postdatedtkt	**yes** にセットされます。つまり、アカウントには、将来の開始時間を持つチケットを発行することができます。
forwardabletkt	**yes** にセットされます。つまり、アカウントには、現在のチケット発行チケットとネットワーク・アドレスが異なる、新しいチケット発行チケットを発行することができます。
renewabletkt	**yes** にセットされます。つまり、アカウントのチケットは更新することができます。
proxiabletkt	**yes** にセットされます。つまり、アカウントには、現在のチケットとネットワーク・アドレスが異なるチケットを発行することができます。
dupkey	**yes** にセットされます。つまり、アカウントのチケットは複写キーを持つことができます。
goodsince	アカウントが作成された日付にセットされます。
maxtktlife	レジストリー・ポリシーにセットされます。
maxtktrenew	レジストリー・ポリシーにセットされます。 **maxtktrenew** 属性は、現在 DCE によっては使用されていません。このオプションの使用は現在サポートされていません。

セル間認証アカウントの変更

registry connect コマンドにより作成されたアカウントは、標準の **dcecp account** 操作を使用していつでも変更することができます。ただし、次の注意事項を理解している必要があります。

アカウントの **pwdvalid** 属性を **no** (無効) にセットしてはなりません。標準アカウントの場合、属性を **no** にセットすると、ユーザーは、次にログインするときにパスワードの変更を要求されます。しかし、セル間認証アカウントのパスワードは、2 つのセルの認証サービスによって共用されています。したがって、一方を変更すると、この同期は破壊され、セル間の通信は終了します。認証サービスが共用するパスワードを変更する必要がある場合には、 **registry connect** コマンドを再度実行して、アカウントを作り直し、正しく同期されたパスワードを作成する必要があります。

一般に、セルとの対等関係を切る場合以外は、アカウントまたはアカウントのプリンシパルを削除しないでください。一方のアカウントが削除された場合は、**registry connect** コマンドを実行して両方のアカウントを再作成し、対等関係を復元する必要があります。

OSF DCE 1.2.2 クライアントからのセル間アクセス

OSF 1.2.2 レベルの DCE のベンダーの実装システムには、一般に外部セルにアクセスするための新しい制約事項が実装されています。 DCE クライアントが外部セルにアクセスには、ローカル・レジストリー (**krbtgt/***foreign_cellname*) 内の外部セル用のセル間認証アカウントの **acctvalid** フラグが **yes** に設定されていなければなりません。そうしないと、外部セルへのアクセスが拒否されます。

DCE 3.2 for AIX および Solaris は、OSF 1.2.2 レベルですが、この制約事項を強制することはありません。ただし、OSF DCE 1.2.2 のそれ以外のベンダーの実装システムでは、この制約事項が強制実行されます。セル間アクセスを許可するために、セル管理者は、以下の **dcecp** コマンドを使用して、既存のセル間認証アカウントに対して **acctvalid** フラグを **yes** に変更することができます。

```
dcecp -c account modify krbtgt/ -modify ¥ { acctvalid yes }
```

新しいセル間の関係を確立する場合、セル管理者は、**dcecp registry connect** コマンドの **acctvalid** および **facctvalid** フラグを使用して、セル間認証アカウント上で **acctvalid** フラグを自動的に yes に設定することができます。

第34章 レジストリー情報の表示

dcecp を使用して、次のセキュリティー・オブジェクトについての情報を表示することができます。

- プリンシパル
- グループ
- オーガニゼーション
- アカウント
- レジストリー
- **xattrschema** オブジェクト
- ACL
- keytab ファイル

次の **dcecp** 操作により、上記のものが表示されます。

- **catalog** コマンドは、指定されたすべてのオブジェクトの名前を表示します。
- **list** コマンドは、指定されたグループまたはオーガニゼーションのメンバー名、あるいは指定されたキー・テーブルのメンバー名を表示します。
- **show** コマンドは、オブジェクトの特定のインスタンスについての情報を表示します。

この章では、 457ページの『第36章 ルーチン保守の実行』に記述されているレジストリー・オブジェクトを除く、すべてのセキュリティー・オブジェクトが使用可能な操作を表示する方法について説明します。

アカウント情報の表示

アカウントについての情報を表示するには、 **dcecp account catalog** と **account show** コマンドを使用します。 **account show** コマンドを使用する場合は、表示するアカウントを指定するために、アカウントのプリンシパル名を指定する必要があります。複数のプリンシパル名を指定するときには、それらを中括弧で囲み、それらの間をスペースで区切ります。

名前にセル名を接頭部として付けて、レジストリー・データベースのすべてのアカウントをアルファベット順で表示するには、次のように入力します。

```
account catalog
```

名前にセル名を接頭部として付けずに、レジストリー・データベース内のすべてのアカウントをアルファベット順に表示するには、次のように入力します。

```
account catalog -simplename
```

指名したプリンシパルのアカウントのすべての属性を表示するには、次のように入力します。

```
account show principal_name
```

指名したプリンシパルのアカウントのすべてのポリシーを表示するには、次のように入力します。

account show *principle_name* **-policies**

指名したプリンシパルのアカウントのすべての属性とすべてのポリシーを表示するには、次のように入力します。

account show *principal_name* **-all**

次に示すのは、**-simplename** オプションを指定せずに **account catalog** コマンドを使用した例です。

```
dcecp> account catalog
/.../dresden.com/bach
/.../dresden.com/bin
/.../dresden.com/brahms
/.../dresden.com/britten
/.../dresden.com/cell_admin
/.../dresden.com/daemon
/.../dresden.com/dce-ptgt
/.../dresden.com/dce-rgy
/.../dresden.com/mahler
/.../dresden.com/nobody
/.../dresden.com/root
/.../dresden.com/uucp
/.../dresden.com/hosts/pmin17/cds-server
/.../dresden.com/hosts/pmin17/gda
/.../dresden.com/hosts/pmin17/self
/.../dresden.com/krbtgt/dresden.com
dcecp>
```

次の例は、**account show** コマンドを使用して、アカウント **mahler** に関連付けられている属性を表示しています。

```
dcecp> account show mahler
{acctvalid yes}
{client yes}
{created /.../dresden.com/cell_admin
1994-06-15-18:31:08.000+00:00I-----}
{description {}}
{dupkey no}
{expdate 1995-06-16-00:00:00.000+00:00I-----}
{forwardabletkt yes}
{goodsince 1994-06-15-18:31:05.000+00:00I-----}
{group users}
{home /}
{lastchange /.../dresden.com/cell_admin
1994-06-16-12:21:07.000+00:00I-----}
{organization users}
{postdatedtkt no}
{proxiabletkt no}
{pwdvalid yes}
{renewabletkt yes}
{server yes}
{shell {}}
{stdtgtauth yes}
dcecp>
```

アカウントに定義されたポリシーが、レジストリー・ポリシーにより指定変更されているために実際には有効でない場合、ポリシーの後には、 **effective** タグおよび有効である実際の値が続くことに注意してください。

グループおよびオーガニゼーション情報の表示

グループについての情報を表示するには、**dcecp group catalog**、 **group show**、および **group list** コマンドを使用します。オーガニゼーションについての情報を表示するには、 **dcecp organization catalog**、**organization show**、および **organization list** コマンドを使用します。 **group list**、**group show**、**organization list**、および **organization show** コマンドを使用するときには、表示するグループあるいはオーガニゼーションの名前を指定する必要があります。複数の名前を指定するときには、それらを中括弧で囲み、それらの間をスペースで区切ります。

名前にセル名を接頭部として付け、レジストリー・データベースのすべてのグループまたはオーガニゼーションをアルファベット順で表示するには、次のように入力します。

```
group catalog
```

または、

```
organization catalog
```

名前にセル名を接頭部として*付けずに*、レジストリー・データベースのすべてのグループまたはオーガニゼーションをアルファベット順で表示するには、次のように入力します。

```
group catalog -simplename
```

または、

```
organization catalog -simplename
```

名前にセル名を接頭部として付けて、指定したグループまたはオーガニゼーションのすべてのメンバーをアルファベット順で表示するには、次のように入力します。

```
group list group_name
```

または:

```
organization list organization_name
```

名前にセル名を接頭部として*付けずに*、指定したグループまたはオーガニゼーションのすべてのメンバーをアルファベット順で表示するには、次のように入力します。

```
group list group_name -simplename
```

または:

```
organization list organization_name -simplename
```

グループあるいはオーガニゼーションのすべての属性を表示するには、次のように入力します。

```
group show group_name
```

または:

```
organization show organization_name
```

グループあるいはオーガニゼーションに付加されたすべての拡張属性インスタンスを表示するには、次のように入力します。

group show *group_name* **-xattrs**

または:

organization show *organization_name* **-xattrs**

グループあるいはオーガニゼーションのすべての正規の属性とすべての拡張属性を表示するには、次のように入力します。

group show *group_name* **-all**

または:

organization show *organization_name* **-all**

次に示すのは、**-simplename** オプションを指定せずに **group catalog** コマンドを使用した例です。

```
dcecp> group cat
/.../dresden.com/nogroup
/.../dresden.com/system
/.../dresden.com/daemon
/.../dresden.com/uucp
/.../dresden.com/bin
/.../dresden.com/kmem
/.../dresden.com/mail
/.../dresden.com/tty
/.../dresden.com/none
/.../dresden.com/tcb
/.../dresden.com/acct-admin
/.../dresden.com/subsys/dce/sec-admin
/.../dresden.com/subsys/dce/cds-admin
/.../dresden.com/subsys/dce/dts-admin
/.../dresden.com/subsys/dce/cds-server
/.../dresden.com/subsys/dce/dts-servers
/.../dresden.com/users
dcecp>
```

次の例は、**users_temporary** という名前のグループの属性を示しています。

```
dcecp> group show users_temporary
{alias no}
{gid 5211}
{uuid 0000145b-9362-21cd-a601-0000c08adf56}
{inprojlist no}
{fullname {temporary users}}
dcecp>
```

上記の例には **{alias no}** となっている行があります。これは、名前 **users_temporary** が 1 次名であり、別名ではないことを示しています。別名の場合、この行は **{alias yes}** となります。

次の **group list** コマンドは、グループ **symphonists** のメンバーを表示します。

```
dcecp> group list  symphonists
/.../dresden.com/bach
/.../dresden.com/britten
/.../dresden.com/mahler
dcecp>
```

プリンシパル情報の表示

プリンシパルについての情報を表示するには、 **dcecp principal catalog** と **principal show** コマンドを使用します。 **principal show** コマンドを使用する場合は、表示するプリンシパルの名前を指定する必要があります。複数のプリンシパル名を指定するときには、それらを中括弧で囲み、それらの間をスペースで区切ります。

名前にセル名を接頭部として付けて、レジストリー・データベースのすべてのプリンシパルをアルファベット順で表示するには、次のように入力します。

```
principal catalog
```

名前にセル名を接頭部として付けずに、レジストリー・データベース内のすべてのプリンシパルをアルファベット順に表示するには、次のように入力します。

```
principal catalog -simplename
```

指名したプリンシパルのすべての属性を表示するには、次のように入力します。

```
principal show principal_name
```

プリンシパルに付加されたすべての拡張属性インスタンスを表示するには、次のように入力します。

```
principal show principal_name -xattrs
```

プリンシパルのすべての正規の属性とすべての拡張属性を表示するには、次のように入力します。

```
principal show principal_name -all
```

次に示すのは、**-simplename** オプションを指定して **principal catalog** コマンドを使用した例です。

```
dcecp> principal catalog -simplename
bach
bin
brahms
britten
cell_admin
daemon
dce-ptgt
dce-rgy
mahler
nobody
root
uucp
cds-server
dcecp>
```

次の例は、**principal show** コマンドを使用して、 **mahler** というプリンシパルに関する情報を表示しています。

```
dcecp> principal show /.:/mahler
{fullname {Gustav Mahler}}
{uid 30014}
{uuid 0000753e-f51f-2e0e-b000-0000c08adf56}
{alias no}
{quota unlimited}
{groups {symphonists composers}}
dcecp>
```

principal show コマンドによってリストされる情報はすべて、プリンシパルがレジストリーに追加されたときに作成された情報です (グループの行を除く)。この行は、プリンシパルがメンバーであるグループをリストします。

xattrschema 情報の表示

拡張属性型についての情報を表示するには、 **dcecp xattrschema catalog** と **xattrschema show** コマンドを使用します。プリンシパルに付加されている拡張属性のインスタンスを見るには、 **-xattr** オプションを **principal**、**group**、**organization show** コマンドに指定することに注意してください。

xattrschema catalog コマンドは、指定したスキーマに定義されている拡張属性オブジェクトの名前を表示します。このコマンドを使用する際には、拡張属性を表示するスキーマの名前を指定する必要があります。レジストリー・データベースの場合、この名前は **/.:/sec/xattrschema** です。サイトは、スキーマの名前をこのコマンドの使用者に提供する必要があります。

xattrschema show コマンドは、レジストリー・スキーマ内の、あるいはサイトで使用されているスキーマ内の、指名されたスキーマの属性を表示します。このコマンドを使用する際には、情報を表示する拡張属性型の名前を指定する必要があります。複数の名前を指定するときには、それらを中括弧で囲み、それらの間をスペースで区切ります。

レジストリー・データベース内のすべての属性型の名前を、名前の接頭部にセル名を付けてアルファベット順で表示するには、次のように入力します。

```
xattrschema catalog /.:/sec/xattrschema
```

レジストリー・データベース内のすべての属性型を、名前の接頭部にセル名を付けずに、アルファベット順に表示するには、次のように入力します。

```
xattrschema catalog /.:/sec/xattrschema -simplename
```

レジストリー以外のスキーマの属性を表示するには、**/.:/sec/xattrschema** を他のスキーマの完全指定の名前で置き換えます。

指名された拡張属性型の属性を表示するには、次のように入力します。

```
xattrschema show attr_name
```

次の例は、セル名が接頭部として付いているレジストリーのすべての拡張属性の名前をリストします。

```
dcecp> xattrschema catalog /.:/sec/xattrschema
/.../dresden/sec/xattrschema/pre_auth_req
/.../dresden/sec/xattrschema/pwd_val_type
/.../dresden/sec/xattrschema/pwd_mgmt_binding
/.../dresden/sec/xattrschema/X500_DN
/.../dresden/sec/xattrschema/X500_DSA_Admin
/.../dresden/sec/xattrschema/disable_time_interval
/.../dresden/sec/xattrschema/max_invalid_attempts
/.../dresden/sec/xattrschema/passwd_override
/.../dresden/sec/xattrschema/test_integer
dcecp>
```

次の例は、**test_integer** という名前の拡張レジストリー属性の属性をリストしています。

```
dcecp> xattrschema show /.:/sec/xattrschema/test_integer
{aclmgr {principal {{query r} {update r} {test r} {delete r}}}}
{annotation {test_integer: encoding type integer}}
{applydefs yes}
{encoding integer}
{intercell reject}
{multivalued yes}
{reserved no}
{scope {}}
{trigbind {none {}}}
{trigtype none}
{unique no}
{uuid 5f439154-2af1-11cd-8ec3-080009353559}
dcecp>
```

ACL 情報の表示

指定したオブジェクトの ACL エントリーを表示するには、 **dcecp acl show** コマンドを使用します。このコマンドを使用する際には、 ACL エントリーを表示するオブジェクトの名前を指定する必要があります。複数の名前を指定するときには、それらを中括弧で囲み、それらの間をスペースで区切ります。

このコマンドがオブジェクトの名前を判別できない場合は、オブジェクトの UUID が表示されます。

指定されたオブジェクトの ACL エントリーを表示するには、次のように入力します。

acl show *object_name*

ACL のデフォルト・セルを表示するには、次のように入力します。

acl show *object_name* **-cell**

オブジェクトがサポートする ACL マネージャーを表示するには、次のように入力します。

acl show *object_name* **-managers**

次の例は、**hosts** の名前のオブジェクトの ACL エントリーを表示します。

```
dcecp> acl show /.:/hosts
{unauthenticated r--t---}
{user cell_admin rwdtcia}
{user hosts/absolut/cds-server1 rwdtcia}
{user root rwdtcia}
{group subsys/dce/cds-admin rwdtcia}
{group subsys/dce/cds-server rwdtcia}
{any_other r--t---}
dcecp>
```

keytab 情報の表示

アカウントについての情報を表示するには、**dcecp keytab catalog**、 **keytab list**、および **keytab show** コマンドを使用します。 **keytab catalog** コマンドを使用するとき、 keytab ファイルを表示するホストの名前を指定する必要があります。 **keytab list** あるいは **keytab show** コマンドを使用するときには、 keytab

情報を表示する **dced** オブジェクトの名前を指定する必要があります。複数の名前を指定するときには、それらを中括弧で囲み、それらの間をスペースで区切ります。

指定されたホスト上のすべての keytab ファイルの名前を、名前の接頭部にセル名を付けて表示するには、次のように入力します。

keytab catalog *host_name*

host_name を指定しないと、現行ホスト上の keytab ファイルのリストが表示されます。

指定されたホスト上のすべての keytab ファイルの名前を、接頭部にセル名を*付けない*で 表示するには、次のように入力します。

keytab
catalog *host_name* **-simplename**

指定された keytab ファイルにエントリーが存在するすべてのプリンシパルのリストを表示するには、次のように入力します。

keytab list *file_name*

指定の keytab ファイルにエントリーが存在するすべてのプリンシパルを表示するには、次のように入力します。

keytab show *file_name*

表示される情報は、プリンシパル名だけです。

指定されたキー・ファイルのローカル名を表示するには、次のように入力します。

keytab show *dced_object_name* **-entry**

キーなど、キー・ファイルのすべてのエントリーを表示するには、次のように入力します。

keytab show *dced_object_name* **-members**

次の例は、**svr_3** という名前の keytab ファイル内のエントリーを表示します。

```
dcecp> keytab show /.:/hosts/music/config/keytab/svr_3 -members
{brahms des 1}
{britten plain 3}
{mahler des 2}
dcecp>
```

第35章 ポリシーと属性の保守

レジストリー・ポリシーは、レジストリー全体に設定できる属性です。より細かい制御手段を提供するために、ポリシーは個々のオーガニゼーションやアカウントに設定することもできます。オーガニゼーションのポリシーやアカウントのポリシーは、それらのポリシーがより制限的な場合には、レジストリーのデフォルトのポリシーを指定変更することができます。

レジストリー属性は、レジストリー内に作成されたプリンシパル、グループ、およびオーガニゼーションに適用される属性です。レジストリー属性は、個々のオーガニゼーションやアカウントに対しては設定できません。属性は、UNIX ID に対して使用できる数の範囲とか、暗号化されたパスワードが表示されるかどうか、といったことを規制します。

dcecp registry modify コマンドを使用して、ポリシーと属性の両方をセットすることができます。また、**dcecp organization modify** コマンドや **dcecp account modify** コマンドによって、個々のオーガニゼーションやアカウントのポリシーをセットできます。すべてのコマンドにおいて、セットするポリシーと属性は、**-change** オプションとともに標準 **dcecp** 属性リスト内の属性として指定するか、または属性オプションとして指定します。

この章では、まずポリシーについて説明し、次に属性について説明します。

ポリシー

次のものについてポリシーを設定することができます。

- **dcecp registry modify** コマンドにより、レジストリー全体に設定されるポリシー。特定のオーガニゼーションまたはアカウントに対してさらに厳密なポリシーがセットされない限り、このポリシーはすべてのプリンシパル、グループ、およびオーガニゼーションに適用されます。
- **dcecp organization modify** コマンドを使用して、特定のオーガニゼーションに設定されるポリシー。
- **dcecp account modify** コマンドを使用して、特定のアカウントに設定されるポリシー。

ポリシーには、標準ポリシーと認証ポリシーの 2 種類のポリシーがあります。

標準ポリシー

標準ポリシーは、アカウントやパスワードの存続時間およびパスワード・フォーマットなどを規制します。標準ポリシーは、レジストリーと特定のオーガニゼーションに対して設定できます。設定できる標準ポリシーについて、次の項で説明します。

注: この節で説明しているパスワード・ポリシーの定義を行う以外に、パスワード・フォーマット、パスワードの生成、無効なログインの処理、および ERA をプリンシパルに付加することによる失効パスワードの処理などの領域もさら

に制御することができます。詳細については、357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。

アカウント存続期間

設定したアカウント存続期間は、特定のオーガニゼーションまたはレジストリー全体のアカウントが有効である期間を決定します。その期間が過ぎるとアカウントは無効になり、再び作成しなければならなくなります。

アカウント存続期間を、次の形式の **dcecp acctlife** 属性として定義します。

acctlife {*time* | **unlimited**}

ここで *time* は、アカウントが有効である日数を示す数であり、 **unlimited** は無制限の存続期間を指定します。

アカウント存続期間は、**dcecp account create** コマンドまたは **account modify** コマンドを使用してアカウントの作成または変更を行う際にセットする、アカウント有効期限 (**expdate** 属性) によっても制御されます。アカウント存続期間ポリシーと矛盾するアカウント有効期限を設定すると、より厳格な方の設定値が適用されます。たとえば、標準ポリシー・アカウント存続期間を 40 日に設定し、アカウント有効期限を翌日に設定すると、アカウントは翌日に期限切れになります。これがより厳密な設定であるためです。

注: **dcecp account modify** コマンドを使用して、アカウントを無効とマークする (**acctvalid** 属性) ことによって、より直接的なレベルでアカウントの妥当性を制御することができます。

パスワード存続期間

パスワード存続期間は、特定のオーガニゼーションまたはレジストリー全体としてのアカウント・パスワードが、期限切れになるまでの期間を指定します。

一般に、DCE セキュリティーはパスワードの期限が切れたユーザーのログインを不能にします。しかし、**cell_admin** などのユーザーの場合は、期限切れのパスワードによってセル管理者がシステムからロックアウトされないようにするために、このポリシーを指定変更することができます。これを行うには、プリンシパルに *passwd_override* ERA のインスタンスを付加します。この方法については、357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。

アカウント存続期間は、次の形式で **dcecp pwdlife** 属性として定義します。

pwdlife {*time* | **unlimited**}

ここで *time* は、パスワードが有効である日数を示す数であり、 **unlimited** は無制限の存続期間を指定します。

パスワード有効期限ポリシー (**pwdexpdate** 属性) を使用して、パスワードが期限切れになる日付そのものをセットすることもできます。

パスワード有効期限

パスワードの有効期限は、特定のオーガニゼーションまたはレジストリー全体のアカウント・パスワードが期限切れになる日付そのものをセットします。

一般に、DCE セキュリティーはパスワードの期限が切れたユーザーのログインを不能にします。しかし、**cell_admin** などのユーザーの場合は、期限切れのパスワードによってセル管理者がシステムからロックアウトされないようにするために、このポリシーを指定変更することができます。これを行うには、プリンシパルに *passwd_override* ERA のインスタンスを付加します。この方法については、357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。

パスワードの有効期限は、次の形式で **dcecp pwdexpdate** 属性として定義します。

pwdexpdate {*date* | **none**}

ここで、*date* は *yyyy-mm-dd* フォーマットで示された、パスワードが失効する日です。 **none** は、パスワードに有効期限がないことを指定します。

パスワード存続期間ポリシー (**pwdlife** 属性) によって、パスワードの期限が切れるまでの期間をセットすることもできます。

パスワード・フォーマット

パスワード・フォーマット・ポリシーは、特定のオーガニゼーションまたはレジストリー全体に適用されます。パスワード・フォーマットは次のものを決定します。

- 次の形式で、**dcecp registry modify pwdminlen** 属性によって定義される、パスワードの最小長。

  pwdminlen *integer*

  パスワードを、*integer* に入力された数値より少ない文字で構成することはできません。 **0** (ゼロ) を指定すると、最小長は無効です。
- 次の形式で **dcecp pwdspaces** 属性によって定義された、パスワードがすべてスペースで構成できるかどうかの指定。

  pwdspaces {yes | no}

  no を指定した場合は、パスワードをすべてスペースで構成することはできません。
- 次の形式で **dcecp pwdalpha** 属性によって定義された、パスワードを英数字のみで構成できるかどうかの指定。

  pwdalpha {yes | no}

  no を指定すると、パスワードには少なくとも 1 つの非英数字を含める必要があります。

注: プリンシパルに ERA を付加することによって、さらにパスワード・フォーマットを制御することができます。これを行う方法については、 357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。

認証ポリシー

認証ポリシーは、チケット存続時間を規制します。認証ポリシーの設定は、レジストリー全体に対しては **dcecp registry modify** コマンドを使用し、個々のアカウ

ントに対しては **dcecp account modify** コマンドを使用して行います。設定できる認証ポリシーについては、次の項で説明します。

注: この項で説明する認証ポリシーの他に、*pre_auth_req* ERA のインスタンスをプリンシパルに付加することによって、プリンシパルの事前認証ポリシーも制御できることに注意してください。事前認証の全般的な説明および、事前認証管理については、 357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』を参照してください。

最大チケット更新可能時間

注: DCE RPC は現在、サービス・チケットの更新時に、更新可能チケットの機能を使用していません。しかし、DCE セキュリティー・サーバーはこれをサポートしており、 DCE セキュリティー・サーバーを KDC として使用する Kerberos V5 アプリケーションにとってはこの機能は便利です。

セットされた最大チケット更新可能時間 (**maxtktrenew** 属性) は、プリンシパルのチケット発行チケットが期限切れになる最長時間 (時間単位) であり、再認証して別のチケット発行チケットを入手するために、プリンシパルが再度ログインしなければならない時間を決定します。最大チケット更新可能時間を短くするほど、システムのセキュリティーの度合いは高まります。ただし、ユーザーは再度ログインしてチケット発行チケットを更新しなければならないので、ユーザーの便利さと必要なセキュリティーのレベルを考慮に入れて時間を決める必要があります。

最大チケット更新可能時間は、**dcecp maxtktrenew** 属性を使用して次の形式で定義します。

maxtktrenew *hours*

ここで、*hours* はプリンシパルのチケット発行チケットが期限切れになるまでの時間数を示す数値です。

account modify コマンドを使用して、個々のアカウントに対して、この時間をセットできることに注意してください。

最大チケット存続時間

最大チケット存続時間 (**maxtktlife** 属性) は、プリンシパルに発行されたチケットが有効である最長の時間 (時間単位) です。クライアントがサーバーへのチケットを要求したとき、そのチケットに付与される存続時間は、サーバーとクライアントの両方に設定された最大チケット存続時間を考慮に入れて決められます。付与される存続時間は、サーバーの最大チケット存続時間とクライアントの最大チケット存続時間のいずれか短い方を超えません。

最大チケット存続時間は、**dcecp maxtktlife** 属性を使用して次の形式で定義します。

maxtktlife *hours*

ここで、*hours* はプリンシパルに発行されるチケットが有効である時間数を示す数値です。

最大チケット存続時間を短くするほど、システムのセキュリティーは高まります。しかし、更新が極端に頻繁になると、処理のオーバーヘッドが大きくなります。最

大チケット存続時間の設定には、システム・パフォーマンスおよび、必要とするセキュリティーのレベルを考慮に入れる必要があります。

account modify コマンドを使用して、個々のアカウントに対して、この時間をセットできることに注意してください。

矛盾するポリシーの処理

レジストリー全体、および個々のオーガニゼーション (標準ポリシーの場合) とアカウント (認証ポリシーの場合) に対して、異なる標準ポリシーと認証ポリシーを有効にすることができます。レジストリー全体に対して設定されたポリシーが、個々のオーガニゼーションまたはアカウントに対して設定されたポリシーと異なる場合、より厳格な方のポリシーが適用されます。たとえば、レジストリー・ポリシーが 6 文字の最小パスワード長を指定し、 **classic** という名前のオーガニゼーションのポリシーが 8 文字を指定しているとします。アカウント **bach cantata classic** を作成すると、より厳格な方のポリシー (この場合、オーガニゼーションのポリシー) が適用されるので、アカウント・パスワードの長さは少なくとも 8 文字でなければなりません。 表25 に、各ポリシー・タイプごとの、より厳格なポリシーをリストします。

表 25. より厳格な標準ポリシー

このタイプのポリシーの場合	より厳格なポリシー
パスワード有効期限	期限切れ期間の短い方。
パスワード存続期間	存続期間の短い方。
アカウント存続期間	存続期間の短い方。
パスワード長	より長い方。
全部スペースで構成されるパスワード	パスワードをすべてスペースで構成することはできず、何かの文字を含める必要がある。
すべて英数字で構成されるパスワード	パスワードをすべて英数字で構成することはできず、非英数字を含める必要がある。
最大チケット更新可能	短い方の時間 (注:DCE RPC は現在、サービス・チケットの更新時に、更新可能チケット機能を使用していません。しかし、DCE セキュリティー・サーバーはこれをサポートしており、 DCE セキュリティー・サーバーを KDC として使用する Kerberos V5 アプリケーションにとってはこれは便利です。)
最大チケット存続時間	短い方の時間。

レジストリーが作成される時、標準ポリシーは、デフォルトにより、最も寛大な状態になります。つまり、パスワード有効期限は **none** となり、パスワードとアカウントの存続期間は **unlimited** となり、最小パスワード長は **0** となり、パスワードは全部スペースで構成でき、またすべて英数字で構成できます。最大チケット存続時間は 10 時間にセットされます。 (最大チケット更新可能は現在使用されていません。) より厳格なポリシーを実現するには、**registry modify** コマンドを使用する必要があります。

既存のポリシーに対する変更の影響

449ページの『パスワード・フォーマット』に説明されているパスワード・フォーマット・ポリシーを除き、ポリシーの変更は、既存のすべてのアカウントおよび、その変更以後に作成するすべてのアカウントに影響します。

パスワード・フォーマット・ポリシー (パスワード長、パスワードをすべてスペースで構成できるかどうか、およびパスワードをすべて英数字で構成できるかどうか、など) に対する変更は、そのポリシーが変更された後で作成されるアカウントのパスワードに対してのみ影響します。それらの変更は、既存のパスワードには影響を与えません。たとえば、より長い長さのパスワードを強制するために最小パスワード長ポリシーを変更した場合、新しいポリシーで指定された長さより短い、既存のパスワードは影響を受けません。既存のパスワードを変更する必要はありませんが、新しいパスワードを作成する場合は、新しいポリシーに従う必要があります。ただし、既存のパスワードを次に変更するときは、より長い長さのポリシーが適用されます。

標準ポリシーと認証ポリシーの表示と設定

ポリシーを表示するには、次のように行います。

- レジストリー全体の場合は、**dcecp registry show** コマンドに **-policies** オプションを指定して使用します。
- 個々のオーガニゼーションまたはアカウントの場合、標準ポリシーならば、**dcecp organization show** コマンドに **-policies** オプションを指定して使用し、認証ポリシーならば **dcecp account show** コマンドに **-policies** オプションを指定して使用します。

ポリシーを設定するには、次のように行います。

- レジストリー全体の場合は、**dcecp registry modify** コマンドを使用します。次のサンプル・コマンドは、レジストリー全体に対してパスワード存続期間ポリシーを 180 日に設定するために、**pwdlife** オプションを使用します。

  ```
  dcecp> registry modify -pwdlife 180
  dcecp>
  ```

- 個々のオーガニゼーションまたはアカウントの場合、標準ポリシーには **dcecp organization modify** コマンドを使用し、認証ポリシーには **dcecp account modify** コマンドを使用します。次のサンプル・コマンドは、オーガニゼーション **classic** のパスワード存続期間ポリシーを **unlimited** にセットするために、**-pwdlife** 属性オプションを使用します。

  ```
  dcecp> organization modify classic -pwdlife unlimited
  dcecp>
  ```

上記の例すべてが属性オプションを使用していることに注意してください。**-change** オプションと属性リストを指定した、**dcecp registry modify**、 **dcecp account modify**、および **dcecp organization modify** コマンドを使用しても、ポリシーをセットすることができます。たとえば、属性リストを使用して、オーガニゼーション **classic** のパスワード存続期間ポリシーを **unlimited** にセットする場合、コマンドは次のようになります。

```
dcecp> organization modify classic -change {pwdlife unlimited}
dcecp>
```

属性 (プロパティー)

dcecp registry modify コマンドは、レジストリー全体の属性をセットします。セットできる属性については、次の項で説明します。

デフォルトのチケット存続時間属性

デフォルトのチケット存続時間は、レジストリー内のプリンシパルに対して発行されるチケットのデフォルトの存続時間 (時間単位) です。

デフォルトのチケット存続時間は、次の形式で **dcecp deftktlife** 属性を使用してセットします。

deftktlife *hours*

ここで、*hours* は存続時間の時間数を示す数値です。

隠しパスワード属性

隠しパスワード属性は、暗号化されたパスワードを表示するかどうかを決めます。隠しパスワード属性は、**dcecp hidepwd** 属性を使用して、次の形式でセットします。

hidepwd
{yes | no}

ここで **yes** にセットすると、コマンド出力、およびパスワードが表示されるファイルにおいて、暗号化されたパスワードの代わりに ***** (アスタリスク) が表示され、**no** にセットすると、隠しパスワードが表示されます。

最小グループ ID 属性

最小グループ ID 属性は、グループのアカウントがレジストリーに追加されるときに、DCE セキュリティー・サービスによって自動的に生成されるグループ ID の開始点です。 (この数値より低いグループ ID を明示的に入力することができます。それは、自動的に生成された番号にのみ適用されます。)

最小グループ ID 属性は、**dcecp mingid** 属性を使用して、次の形式でセットします。

mingid *integer*

ここで *integer* は、開始の ID 番号です。

最小オーガニゼーション ID 属性

最小オーガニゼーション ID 属性は、オーガニゼーションのアカウントがレジストリーに追加されるときに、セキュリティー・サービスが自動的に生成するオーガニゼーション ID の開始点です。 (この数値より低いオーガニゼーション ID を明示的に入力することができます。その設定は、自動的に生成される番号にのみ適用されます。)

最小オーガニゼーション ID 属性は、**dcecp minorgid** 属性を使用して、次の形式でセットします。

minorgid *integer*

ここで *integer* は、開始の ID 番号です。

最小 UNIX ID 属性

最小 UNIX ID 属性は、プリンシパルのアカウントがレジストリーに追加されるときに、セキュリティー・サービスが自動的に生成する UNIX ID の開始点です。(この数値より低い UNIX ID を明示的に入力することができます。その設定は、自動的に生成される番号にのみ適用されます。)

最小 UNIX ID 属性は、**dcecp minuid** 属性を使用して、次の形式でセットします。

minuid *integer*

ここで *integer* は、開始の ID 番号です。

最大 UNIX ID 属性

最大 UNIX ID 属性 (**maxuid** 属性) により、プリンシパルのアカウントが作成される時に、 UNIX ID として指定できる最高の数を設定することが可能となります。この最大数は、システム生成の UNIX ID とユーザー入力の UNIX ID の両方に適用されます。

最大 UNIX ID 属性は、 **dcecp maxuid** 属性を使用して、次の形式でセットします。

maxuid *integer*

ここで *integer* は、開始の UNIX ID です。

最小チケット存続時間属性

最小チケット存続時間は、プリンシパルのチケットを更新する必要が生じるまでの最短時間 (分単位) です。この更新は、ユーザー側の介入なしに、自動的に行われます。最小チケット存続時間を短くするほど、システムのセキュリティーは高まります。しかし、更新を極端に頻繁に行うと、システム・パフォーマンスが低下します。最小チケット存続時間を設定する際には、システム・パフォーマンスと、セルが必要とするセキュリティーのレベルを考慮に入れる必要があります。

最小チケット存続時間属性は、**dcecp mintktlife** 属性を使用して、次の形式でセットします。

mintktlife *integer*

ここで *integer* は、最小チケット存続時間の分数を示す数値です。

最小チケット存続時間は、レジストリー属性としてのみ設定することができます。個々のアカウントに対して設定することはできません。 (これは、**dcecp registry modify** や **account modify** コマンドでセットされる最大チケット存続時間属性とは対照的です。)

属性の表示と設定

レジストリー属性を表示するには、**dcecp registry show** コマンドを使用します。

レジストリー属性をセットするには、**dcecp registry modify** コマンドを使用します。次の例は、**maxuid** オプションを使用して、最大 UNIX ID 属性を 67899 に変更しています。

```
dcecp> registry modify -maxuid 67899
dcecp>
```

上記の例では属性オプションを使用していることに注意してください。 **-change** オプションと属性リストを指定した、 **dcecp registry modify** コマンドを使用しても属性をセットできます。たとえば、属性リストを使用して最大 UNIX ID を 67899 に設定する場合、コマンドは次のようになります。

```
dcecp> registry modify -change {maxuid 67899}
dcecp>
```

第36章 ルーチン保守の実行

この章では、次のような定期的に行う必要があるセキュリティー保守手順について説明します。

- レジストリーへの新規ユーザーの追加
- 個々のマシンに対するオーバーライドの作成
- マスター・キーの変更
- データベースのバックアップと復元
- レジストリーとの整合性のための、**/etc/passwd** および **/etc/group** ファイルの更新

アカウントの追加

新規ユーザー・アカウントをレジストリーに追加するためには、そのレジストリーに対する適切な許可が必要です (533ページの『第43章 レジストリー・オブジェクトへのアクセス』を参照してください)。適切な許可を持っていれば、次のようにアカウントの追加に進むことができます。

1. アカウントで使用するプリンシパルがまだ存在していない場合には、プリンシパルを追加するために **principal create** コマンドを実行する。
2. このグループがまだ存在していない場合には、アカウントで使用するグループを追加するために **group create** コマンドを実行する。
3. このオーガニゼーションがまだ存在していない場合には、アカウントで使用するオーガニゼーションを追加するために **organization create** コマンドを実行する。
4. 最後に、アカウントを追加するために **account create** コマンドを実行する。

ローカル・レジストリーにおけるエントリーのオーバーライド

ローカル・マシンのレジストリー・エントリーをオーバーライドすることができます。オーバーライドを使用すると、たとえば、個人やグループが特定のマシンにログインできないようにしたり、ローカル・ルート・パスワードを設定したり、ローカル・ユーザーの環境を調整したりすることができます。オーバーライド情報はローカル・マシンに対してのみ有効で、レジストリーに格納されているアカウント情報に対しては効果がありません。

オーバーライド・メカニズムにより、高水準のローカルの自主性が与えられ、個々のユーザーが自分のマシンを制御することが可能となります。たとえば、マシンのグループに責任を持つ管理者がオーバーライド機能を使用して、これらのマシンへのアクセスを制限することができます。管理者は、特定のグループへのアクセスを許可したり、特定のグループまたはプリンシパルを除く、すべてへアクセスすることを許可したりすることができます。

オーバーライドの機能

ローカル・マシンの **/opt/dcelocal/etc** ディレクトリーに格納されている、**passwd_override** 管理ファイルの中にオーバーライド情報が入っています。このファイルを使用すれば、以下のものに対してオーバーライドを入力することができます。

- パスワード
- GECOS 情報
- ホーム・ディレクトリー
- ログイン・シェル
- グループ・メンバーシップ
- プリンシパルの UNIX ID

入力したオーバーライド情報は、 **passwd_override** ファイルが格納されているマシンである、ローカル・マシンに対してのみ有効です。ユーザーがオーバーライド・ファイルを使用してマシンにログインすると、オーバーライド・ファイルにあるユーザーのアカウントに対するすべての情報が、レジストリーから取得された該当する情報に置き換わります。

たとえば、**bach** のレジストリー・アカウントが、ログイン時に Korn シェルを指定したとします。 **bach** は、通常、Korn シェルを実行できるマシンにログインするので、多くの場合はこの指定は妥当です。しかし、 **bach** は、時々、別の部門のために働き、Korn シェルを実行できないマシンにログインします。 **bach** のニーズを満たすために、 Korn シェルを実行できないマシン上にオーバーライド・ファイルを作成することができます。オーバーライドは Bourne ログイン・シェルを指定することができます。それにより、**bach** が Korn シェルを実行できるマシンにログインすると、レジストリー・データが使用され、Korn シェルが起動されます。**bach** が Korn シェルを実行できないマシンにログインすると、オーバーライド・データが使用され、Bourne シェルが起動されます。

注: オーバーライド・サービスは、 DCE スリム・クライアント・マシンでは使用することができません。オーバーライド・サービスは、**dced** デーモンにより提供されますが、これはスリム・クライアントでは構成または実行することができません。

passwd_override ファイル・フォーマット

passwd_override ファイルのエントリーのフォーマットは次のとおりです。

principal_name:*passwd*:*principal_uid*:*group_uid*:*GECOS*:*home_dir*:*shell*

ここで、

principal_name
: プリンシパル名が入るキー・フィールド。プリンシパル名は、オーバーライドが適用されるアカウントを持つプリンシパルを識別します。プリンシパルの 1 次名のアカウントにだけオーバーライドを適用し、プリンシパルの別名のアカウントには適用しない場合は、 *principal_name* を入力します。

  オーバーライドが適用されるアカウントを識別するために、キー・フィールド (*principal_name*、*principal_uid*、または *group_uid*) のいずれかを入力する必要があります。

passwd 暗号化されたパスワード。オーバーライドを指定した場合、ここで入力するパスワードはこのローカル・マシンだけに有効です。

passwd フィールドに **OMIT** と指定して、ローカル・マシンでのログインを不許可にすることもできます。 **passwd_export** コマンドのオプションとともに **OMIT** を使用すると、 **passwd_export** コマンドによって作成されたパスワード・ファイルに、このユーザーが含まれなくなります。 (464ページの『ローカル・パスワード・ファイルからのユーザーの省略』を参照。)

principal_uid

principal_uid フィールドには、プリンシパルのローカル識別を指定する UNIX ID が入ります。このフィールドは、*principal_name* キー・フィールドが入力されない場合は、キー・フィールドとして働き、*principal_name* とともに入力された場合は、オーバーライドを含んだフィールドとして働きます。

プリンシパルのアカウントすべて (プリンシパルの別名のアカウントを含む) にオーバーライドを適用したい場合は、 *principal_uid* を入力し、*principal_name* は入力しません。 *principal_uid* キー・フィールドは、特に **root** に対するオーバーライドの場合に有用です。たとえば、**root** に別名 **virtuoso** がある場合、プリンシパル名をキーとするオーバーライドは、ルートが **root** としてログインした場合だけ適用されます。ルートの *principal_uid* をキーとするオーバーライドは、ルートが **root** として、**virtuoso** として、および他のいずれかの別名のもとでログインしたときに、適用されます。

指定されたプリンシパルの UNIX ID をオーバーライドするためには、*principal_uid* と *principal_name* を入力します。

group_uid

group_uid フィールドには、グループの UNIX ID が入ります。このフィールドは、他のキー・フィールドが入力されない場合は、キー・フィールドとして働き、 *principal_name* または *principal_uid* とともに入力された場合は、オーバーライドを含んだフィールドとして働きます。

group_uid で指定したグループのすべてのメンバーにオーバーライドを適用する場合は、 *group_uid* を入力し、他のキー・フィールド (*principal_name* または *principal_uid*) は入力しません。この場合は、*group_uid* フィールドはキー・フィールドとして働き、オーバーライドを適用するアカウント、つまり、そのプリンシパルが、指定されたグループのメンバーである、アカウントを識別します。

principal_name で指定したプリンシパルのグループを、 *group_uid* で指定したグループに変更する場合は、 *group_uid* と *principal_name* を入力します。この変更はプリンシパルの 1 次名のアカウントにだけ適用され、プリンシパルの別名のアカウントには適用されません。プリンシパルのアカウントすべて (プリンシパルの別名のアカウントも含む) にグループ・オーバーライドを適用する場合は、 *group_uid* と *principal_uid* を入力します。このような場合には、 *group_uid* フィールドはキー・フィールドとしてではなく、オーバーライド情報を提供するフィールドとして働きます。

GECOS
アカウントの GECOS フィールド。オーバーライドを指定すると、 UNIX の **finger** コマンドによって表示される情報にオーバーライドが反映されます。

home_dir
アカウントのホーム・ディレクトリー。オーバーライドを指定する場合、指定できるディレクトリーはこのマシンにあるアカウントのホーム・ディレクトリーだけです。

shell アカウントのログイン時に起動されるシェル。オーバーライドを指定すると、このマシンへのログイン時に、指定したシェルが起動されます。

group_override ファイル・フォーマット

このセクションでは、**group_override** について説明します。

説明

/opt/dcelocal/etc/group_override 管理ファイルを使用すれば、 **passwd_override** ファイルを用いてネットワーク・レジストリー・データベースの情報をオーバーライドする場合と類似して方法で、グループの UNIX グループ ID をオーバーライドすることが可能となります。

group_override ファイルは、それぞれのマシンに格納されています。このファイルに対して行った変更は、ローカル・マシンにだけ有効であり、中央のレジストリーには影響を及ぼしません。ファイル **group_override** の使用は、レジストリーで提供されているデフォルト・グループ定義がユーザーのローカル UNIX システムに合わない場合に、それらのデフォルト・グループ定義をオーバーライドする際に特に便利です。

ファイル・フォーマット

ファイル **group_override** のエントリーのフォーマットは、 UNIX の **group** ファイルのエントリーのフォーマットと類似しています。このフォーマットは次のとおりです。

```
group_name:passwd:group_uid:members
```

このエントリーでは、*group_name* と *group_uid* がキー・フィールドです。オーバーライドを適用するグループを識別するために、いずれかを入力する必要があります。キー・フィールドは、**passwd_export** コマンドを使用する際に、オーバーライド・ファイル内で検索を行うために使用されます。検索は、オーバーライド・ファイルで指定されているエントリーの順序にしたがって、最初はグループ名、次にグループ UNIX ID の順番で行われます。オーバーライド・エントリーに両方のキー・フィールドを指定すると、グループ名が検索キーとして使用され、後続のフィールドはオーバーライドとして使用されます。

フィールドの説明

以下に、ファイル **group_override** の各エントリーについて説明します。

group_name
オーバーライドが適用されるグループを識別する、名前が入るキー・フィールド。

passwd このフィールドは、暗号化されたパスワードを指定します。このフィールドにオーバーライドを指定すると、入力するパスワードは、このローカル・マシンだけに有効です。

passwd_export コマンドのオプションとともに **OMIT** を使用すると、**passwd_export** コマンドによって作成されたグループ・ファイルに、このグループが含まれなくなります。それにより、 **newgrp** コマンドがローカル・マシン上のこのグループに事実上、許可されなくなります。 (詳細については 『OMIT の使用』を参照。)

group_uid

UNIX グループ ID。このフィールドは、他のキー・フィールドが入力されていない場合、キー・フィールドとして働きます。また *group_name* とともに入力した場合は、オーバーライドを含むフィールドとしても働きます。*group_uid* 値は、ネットワーク・レジストリー・サーバーによって提供された、グループ ID のローカル・オーバーライドを指定します。

members

このフィールドには、グループのメンバーのリストをコンマで区切って指定します。このフィールドの内容は、**passwd_export** コマンドが **/etc/group** ファイルを作成するときに、レジストリーの情報をオーバーライドします。ただし、ヌル・メンバーシップを指定する場合は、オーバーライドが必要ないことを示すのと区別するために、このフィールドに ***** (アスタリスク) を使用するので注意してください。

ブランクのままのフィールド

項目をオーバーライドしたくない場合は、そのフィールドをブランクのままにして、それぞれのブランク・フィールドを : (コロン) で区切ります。グループをヌル・メンバーシップ・リストでオーバーライドする場合は、 *members* フィールドにアスタリスクを入力することに注意してください。

OMIT の使用

OMIT を指定して、 **-x** オプションを指定した **passwd_export** コマンドを実行すると、名前を指定されたグループは、 **passwd_export** コマンドによって作成された **/etc/group** ファイルには現れません。この後、ユーザーはローカル・マシン上のこのグループに対しては、 **newgrp** コマンドを実行することはできません。

ls コマンドも同様に影響を受けます。たとえば、次のコマンドはグループ・ファイルにアクセスして、グループについての追加情報を取得します。

```
ls -lg
```

グループが省略されると、グループ・エントリーが存在しないので、情報は入手できません。このため、**OMIT** を使用してグループをファイル **/etc/group** から省略するのは、ユーザー・コミュニティーが非常に大きく、次の条件のいずれかが発生した場合だけに限定する必要があります。

- グループ・ファイルがスペースを取りすぎている。
- グループの ID から名前へのマッピングが遅すぎる (たとえば、**ls -lg** コマンドの間)。

例

グループ **kmem** のグループ ID をオーバーライドして **3** にするために、次のエントリーを使用します。

```
kmem::3:
```

グループ **system** のグループ・パスワードとメンバーシップをオーバーライドして、単一のアカウント **root** にするためには、次のエントリーを使用します。

```
system:*::root
```

オーバーライド・ファイル・エントリーの作成

オーバーライド・ファイル・エントリーを作成するためには、 **passwd_override** ファイルを編集して、オーバーライド・エントリーを指定します。エントリーは、以下のキー・フィールドのいずれかを指定することにより、オーバーライドを適用するアカウント (1 つまたは複数) を識別する必要があります。

principal_name
: オーバーライドを適用する特定のプリンシパルの名前。オーバーライドは、プリンシパルの 1 次名のアカウントにのみ適用されます。たとえば、オーバーライド・エントリーの 1 次名として **mahler** を指定すると、オーバーライドはプリンシパル **mahler** のアカウントにだけ適用され、 **mahler** の別名のアカウントには適用されません。

principal_uid
: *principal_name* を指定しない場合にオーバーライドを適用する、アカウントを識別する UNIX ID。オーバーライドは、*principal_uid* によって識別される、プリンシパルのアカウントすべて (プリンシパルの別名のアカウントを含む) に適用されます。たとえば、プリンシパル **mahler** が、 **2195** という UNIX ID を持っているものとします。エントリーのキーとして **2195** を指定すると、オーバーライドはその UNIX ID に関連したアカウントすべてに適用されます。プリンシパルの 1 次名と別名は同じ UNIX ID を持っているため、これはオーバーライドがプリンシパルの 1 次名とすべての別名のアカウントに適用されることを意味します。

group_uid
: *principal_name* も *principal_uid* も指定しない場合に、オーバーライドを適用するグループを識別する UNIX ID。オーバーライドは、識別されたグループのメンバーである、すべてのプリンシパルのすべてのアカウントに適用されます。

principal_name フィールドは常にキー・フィールドとして働き、オーバーライドすることはできません。 *principal_name* を入力すると、それはオーバーライドする特定のアカウントを識別します。 *principal_uid* または *group_uid* フィールドは、キー・フィールドとしても、オーバーライド・フィールドとしても働きます。可能なキー・フィールドのうち 1 つだけがエントリーのキーとして使用され、残り (入力されていれば) は、オーバーライド・フィールドとして使用されます。キーとしてはまず *principal_name* フィールドが優先して使用され、次に *principal_uid* フィールド、最後に *group_uid* の順になります。

たとえば、*principal_uid* を入力して、*principal_name* を入力しない 場合、*principal_uid* がキー・フィールドとして使用されます。 *principal_uid* と

principal_name を入力した場合、 *principal_name* がキー・フィールドとして使用され、 *principal_uid* がオーバーライド・フィールドとして使用されます。

group_uid フィールドを入力して、その他のキー・フィールド (*principal_name* または *principal_uid*) を何も入力しない と、 *group_uid* がキー・フィールドとして使用され、オーバーライドはグループのすべてのメンバーのアカウントに適用されます。 *principal_name* と *group_uid* を入力した場合は、 *principal_name* がキー・フィールドとして使用され、指名されたプリンシパルの加入グループは、グループ UNIX ID で識別されるグループによってオーバーライドされます。

ブランクのままの passwd_override ファイル・フィールド

passwd_override ファイルの項目をオーバーライドしたくない場合、フィールドをブランクのままにして、それぞれのブランク・フィールドを : (コロン) で区切ります。ただし、オーバーライドを作成するプリンシパルまたはグループを識別するために、キー・フィールドの 1 つを入力しなければなりません。たとえば、 **mozart** で識別されるアカウントのホーム・ディレクトリーをオーバーライドするためのエントリーは、次のようになります。

```
mozart:::::/aria/wolfgang:
```

ブランクの後続フィールドに関連したコロンを入力する必要があります。上記の例では、シェル・フィールドに対してコロンが必要です。シェル・フィールドは、ホーム・ディレクトリー・フィールドの後の残りのフィールドです。

特定のマシンに対するパスワードの指定

ローカル・マシンのパスワードをオーバーライドするためにオーバーライド・ファイルにエントリーを作成するには、ローカル・マシンの **passwd_override** ファイルのパスワード・エントリーを手作業で編集します。入力するパスワードは暗号化する必要がありますが、 **/etc/passwd** ファイルから暗号化されたパスワードをコピーすることも、あるいは暗号化されたパスワードを生成するプログラムを書くこともできます。

プリンシパルのパスワードをオーバーライドした場合、ログインで得られるのはプリンシパルのローカル証明書のみであり、プリンシパルのネットワーク証明書は得られません。ネットワーク証明書がなければ、プリンシパルはネットワーク・レジストリーにアクセスできず、ネットワーク・ログイン時に通常提供される情報を入手することができません。したがって、この情報をすべて **password_override** ファイル・エントリーに用意する必要があります。パスワードのオーバーライドについて、オーバーライド・エントリー内のすべてのフィールド (すべてのキー・フィールドを含む) を入力する必要があります。

次の例は、ユーザー **mozart** のアカウントの特定のマシンのパスワードを変更する、 **passwd_override** ファイル・エントリーを示します。

```
mozart:sq1Rc1Urrb1L6:678:893:Wolfgang A. Mozart:/aria/wolfgang:/bin/csh
```

注: パスワードがオーバーライドされた後、 **rlogin** または **rsh** を用いてオーバーライドされたマシンにリモートでログインする場合、パスワードを入力するようにプロンプト指示されます。これは、 **/etc/hosts.equiv** ファイルまたは **.rhosts** ファイルの内容には関係なく、プロンプトが出されます。

マシンへのログインの禁止

ユーザーがマシンにログインできないようにするためには、 **passwd** フィールドに無効なストリングを用いてオーバーライド・エントリーを作成します。 **passwd** フィールドには暗号化されたパスワードが入っているため、長さがちょうど 13 文字でない文字ストリングならば、すべて無効なパスワードとして使用できます。たとえば、**passwd_override** ファイルの以下のエントリーは、パスワードとして **exclude** を提供しています。ストリングが 13 文字以下なので、 UNIX ID **25** で識別されるグループのメンバーはログインできません。

```
:exclude::25:::
```

ローカル・パスワード・ファイルからのユーザーの省略

passwd_override ファイルの無効なパスワード・エントリーは、そのファイルが存在するマシンへユーザーがログインするのを禁止します。しかし、無効なエントリー **OMIT** には特別な意味があります。他の無効なパスワードと同様に、**OMIT** を入力すると、ユーザーはログインできません。さらに、標準の **/etc/passwd** ファイルと **/etc/group** ファイルを保守し、しかも **passwd_export** コマンドを使用してレジストリー・データベースとこれらのファイルの整合性を保っている場合は、パスワード **OMIT** を持つユーザーを **/etc/passwd** ファイルから除外するように指定することができます。 (**passwd_export** コマンドに関する詳細については、 469ページの『整合性のあるローカル・ファイルの保証』を参照してください。)

また、**/etc/passwd** ファイルからユーザーを省略すると、パスワード・ファイルを使用するすべてのプログラムがそのユーザーに関する情報を利用できなくなることにも注意してください。たとえば、 **ls -l** および **finger** コマンドはともに、 UNIX ID が識別するユーザーに関するより詳細な情報を取得するために、パスワード・ファイルをアクセスします。ユーザーを省略すると、パスワード・エントリーが存在しないので、そのユーザーに関する情報は利用できなくなります。

マシンのホーム・ディレクトリーとログイン・シェルの指定

アカウントの特定のマシンのホーム・ディレクトリーとログイン・シェルを変更するためには、ホーム・ディレクトリー名とログイン・シェル名を指定するオーバーライド・エントリーを作成します。たとえば、次のエントリーは、ユーザー **mozart** のアカウントのホーム・ディレクトリーとログイン・シェルを変更します。

```
mozart:::::rondo/mozart:/bin/ksh
```

プリンシパルの加入グループのオーバーライド

プリンシパルの加入グループをオーバーライドするには、プリンシパルの名前または UNIX ID をキーとして含み、また、使用するグループの UNIX ID をオーバーライドとして含んでいる、オーバーライド・エントリーを作成します。オーバーライドをプリンシパルの 1 次名のアカウントにのみ適用する場合は、プリンシパルの名前をキーとして使用します。オーバーライドをプリンシパルのアカウントすべて (プリンシパルの別名のアカウントを含む) に適用する場合は、プリンシパルの UNIX ID をキーとして使用します。たとえば、以下のエントリーは、プリンシパル **mozart** のアカウントに、通常、関連するグループをオーバーライドします。

```
mozart:::356:::
```

このオーバーライドは、 **mozart** の別名のアカウントには適用されません。オーバーライドをそれらのアカウントに適用するためには、以下に示すように、エントリーは **mozart** の UNIX ID (**567**) がキーになっていなければなりません。

```
::567:356:::
```

グループのすべてのメンバーへのオーバーライドの適用

特定のグループのメンバーすべてにオーバーライドを適用するためには、キーとするグループの UNIX ID およびオーバーライドする項目を含んだ、オーバーライド・エントリーを作成します。グループの場合は、パスワード、GECOS 情報、ホーム・ディレクトリー、およびシェルをオーバーライドできます。たとえば、以下のエントリーは、**/sonata/piano** を、 UNIX ID **356** によって識別されるグループのメンバーすべてのホーム・ディレクトリーにします。

```
:::356::/sonata/piano:
```

principal_name または *principal_uid* キー・フィールドを入力しないようにしてください。入力すると、指定したプリンシパル名または UNIX ID がキー・フィールドとして使用され、グループ UNIX ID がそのプリンシパルの加入グループをオーバーライドするために使用されます。

passwd_override が複数のオーバーライド・エントリーを処理する方法

アカウントに複数のオーバーライド・エントリーが適用される場合、最も具体的なアカウント ID (つまり、プリンシパル UNIX ID、グループ UNIX ID、またはプリンシパル名のいずれか) を指定したエントリーが選択されます。プリンシパル名が最も具体的で、その後に、プリンシパル UNIX ID と グループ UNIX ID が続きます。

たとえば、ログイン・シェルをオーバーライドする、以下の 2 つのエントリーがオーバーライド・ファイルに格納されているとします。

```
mozart::::::/bin/ksh
:::25:::/bin/csh
```

プリンシパルが **mozart** としてログインすると、 **mozart** をキーとするオーバーライドが有効になります。このケースでは、プリンシパル (**mozart**) はグループ (**25**) よりも具体的です。

レジストリーのマスター・キーの変更

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

レジストリーに格納されているすべてのパスワードは、マスター・キーによって暗号化されます。システムの構成時にレジストリー・データベースを作成するときに、マスター・キーが作成されることに注意してください。

レジストリーのマスター・キーを変更し、その新規マスター・キーによってすべてのパスワードを再度暗号化するには、 **-key** オプションを指定した **dcecp registry**

modify コマンドを使用することができます。各レプリカ (マスターとスレーブ) は、レジストリー内のコピーの中のデータにアクセスするために、それ自体のマスター・キーを保持しています。

各レプリカのマスター・キーを定期的に変更する必要があります。これを行うためのいずれかのプログラムを実行する前に、管理アカウントにログインしていることを確認してください。

次のコマンド行は、マスター・キーを変更して、レプリカ **art_server_1** のすべてのパスワードを再暗号化します。

```
dcecp> registry modify /.../giverny.com/subsys/dce/sec/art_server_1
-key
dcecp>
```

DCE セキュリティー・サービスの認証の妥当性検査

DCE デーモン内の **secval** プロセスは、 DCE セキュリティー・サーバーが認証を受けたサーバーであることを確認することができます。不正な DCE セキュリティー・サーバーは、偽のローカル・システム識別を戻すことによって、悪意のユーザーにマシン上のルート・アクセスを与えることも可能です。 **secval ping** オペレーションは、 **secval** プロセスに対して認証 RPC を実行することにより、 DCE セキュリティー・サーバーの認証を確認します。正常な戻り値 **1** は、認証 RPC が成功するために必要な、正しいパスワードのすべてをセキュリティー・サーバーが使用したことを示します。

secval ping オペレーションはローカル・ホストで実行することも、引き数を与えてリモート・ホストで動作させることもできます。リモート・ホストは種々のセキュリティー・サーバーを使用することがあるために、 **secval ping** オペレーションをリモート・ホスト上で実行することにより、セルで作動するその他のセキュリティー・サーバーの認証をテストすることができます。

次の例は、リモート・ホスト **charon** 上での、 **secval** プロセスに対する **secval ping** オペレーションを示します。

```
dcecp> secval ping /.:/hosts/charon/config/secval
1
dcecp>
```

レジストリー・データベースのバックアップと復元

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

バックアップ中に、バックアップ内容がマスターに書き込まれないようにしてレジストリー・データベースをバックアップするためには、ここで説明されているとおりの手順を使用してください。

マスター・レプリカ・データベースとそのマスター・キー・ファイルのみをバックアップする必要があります。マスター・レプリカとそのマスター・キーが格納されているディスク全体をバックアップする場合、およびマスターのデータベース・フ

ファイルとそのマスター・キー・ファイルのみをバックアップする場合には、次のサブセクションで説明する手順を使用してください。

レジストリー・データベースをバックアップするための手順

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

バックアップ手順を実行するためには、管理アカウントを用いて DCE にログインする必要があります。その後、DCE 制御プログラムを実行してバックアップを行います。バックアップのステップは次のとおりです。

1. マスター・レプリカを保守状態に設定するために、 **registry disable** コマンドを入力する。次のコマンドは、セル **giverny.com** の中のマスター・レジストリーを保守状態に設定します。

   ```
   dcecp> registry disable  /.../giverny.com/subsys/dce/sec/oddball
   dcecp>
   ```

   マスター・レプリカを保守状態に設定すると、マスターは、そのデータベースをディスクに保管し、すべての更新を拒絶します。
2. ボリューム全体、または *dcelocal***/var/security/rgy_data** ツリー (レジストリー)、およびレジストリーのすべてのキーの暗号化に使用されるマスター・キーを含んだファイルである、 *dcelocal***/var/security/.mkey** ファイルをバックアップすることによって、マスター・レジストリーをバックアップする。ファイル *dcelocal***/var/security/.mkey** はマスター・キーを含んでいるため、レジストリー・データベースを復元しても、ファイル *dcelocal***/var/security/.mkey** も復元しなければ無駄であるため、注意してください。

   バックアップにどのようなコマンドを使用すればよいかは、個人の好みの問題です。しかし、データベースとマスター・キー・ファイルの両方を同じテープに書き込む場合には、そのテープは、アクセスが制限された、ロックした場所に格納してください。あるいは、データベースとキー・ファイルを別々のテープに書き込み、それぞれのテープを別の場所に格納することもできます。
3. バックアップが完了したら、次のようにマスター・レプリカの保守状態を解除します。

   ```
   dcecp> registry enable  /.../giverny.com/subsys/dce/sec/oddball
   dcecp>
   ```

   セキュリティー・サーバーは、更新の受け入れを再開します。

前の例では、**registry enable** コマンドと **registry disable** コマンドにレジストリー・マスター・サイトの名前を指定していることに注意してください。レジストリー・サイト名を指定しない場合は、コマンドは変数 **_s(sec_)** で指定されたサイトを使用します。この変数を設定しなければ、コマンドは、マシンのデフォルトのセルのマスター・レジストリーを使用します。詳細については 469ページの『_s(sec) 変数の設定』を参照してください。

レジストリー・データベースを復元するための手順

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

このセクションでは、マスター・レプリカのデータベース・ファイルおよびマスター・キー・ファイルを復元するための手順について解説します。この手順は、データベースはバックアップされたときと同じマシンに復元されるものと想定し、また、DCE 制御プログラムが使用されるものと想定します。データベースを別のマシンに移動する場合には、 471ページの『第37章 ネットワーク再構成の処理』の指示に従ってください。

レジストリー・データベースをマシンに復元するためには、次のステップにしたがってください。

1. マスター・レジストリー・サイトで **root** としてログインする。
2. **secd** が稼働している場合には、 **registry stop** コマンドを出して、それを停止させる。このコマンドを使用する場合には、特定のレプリカの完全修飾名を引き数として指定する必要があります。次のサンプル・コマンドは、 **oddball** という名前の **secd** を停止させます。

   ```
   dcecp> registry stop /.../giverny.com/subsys/dce/sec/oddball
   dcecp>
   ```

3. バックアップ・ファイルをバックアップ・メディアからマシンにコピーする。レジストリー・データ・ファイルとマスター・キー・ファイルのみをバックアップした場合は、必ずレジストリー・データベースを *dcelocal*/var/security/rgy_data にコピーし、マスター・キー・ファイルを *dcelocal*/var/security/.mkey にコピーしてください。ファイル *dcelocal*/var/security/.mkey はマスター・キーを含んでいるため、レジストリー・データベースを復元しても、ファイル *dcelocal*/var/security/.mkey も復元しなければ無駄であるため、注意してください。
4. 次のように、**-restore_master** オプションを指定した **secd** を呼び出すことによって、サーバーを再スタートする。

   ```
   dcelocal/bin/secd -restore_master&
   ```

   このコマンドにより **secd** が始動し、マスターは再初期設定するすべてのスレーブにマークを付けます。
5. **secd** が、システム始動で自動的に開始することを確認する。

注: マスター・キー・ファイルのみを 復元し、マスター・キーを変更しなかった場合には、復元手順に含まれているその他のステップのすべてを実行しなくても、パックアップ・メディアからマスター・キー・ファイルを簡単にコピーすることができます。

_s(sec) 変数の設定

レジストリーを操作する **dcecp** コマンドに対する引き数として、バインド先のレジストリー・サイトの名前を指定することができます。名前を指定しなければ、コマンドは **_s(sec)** 変数で指定されたレプリカに、バインドします。この変数を設定しなければ、コマンドは、セルのマスター・レプリカにバインドします。 **_s(sec)** 変数を設定してから、そのレプリカを **dcecp registry** コマンドのデフォルトのレプリカとして使用することができます。このためには、次のサンプルに示されている **set** コマンドを使用します。このコマンドは、デフォルトのレプリカをセル **giverny.com** の中のマスター・レプリカ (**slave_3** という名前) に設定します。

```
dcecp> set _s(sec) /.../giverny.com/subsys/dce/sec/slave_3
dcecp>
```

set コマンドに対する引き数として指定する新しいデフォルト・レプリカの名前の形式は、次のいずれかにすることができます。

- セル名 (たとえば、**/.../dresden.com**)

  セル名を入力した場合、指定されたセルがデフォルトのセルになります。 DCE 制御プログラムは、指定されたセルの中でバインドするレプリカをランダムに選択し、そのレプリカがデフォルトのレプリカになります。
- レプリカの作成時にレプリカに与えられるグローバル名 (たとえば、**/.../dresden.com/subsys/dce/sec/rs_server_250_2**)

  グローバル名は、特定のセルの中の特定のレプリカを識別します。そのセルがデフォルトのセルになり、そのレプリカがデフォルトのレプリカになります。
- 現行のデフォルトのレプリカのレプリカ・リストに表示されているレプリカ名 (つまり、セル相対名。たとえば、**subsys/dce/sec/rs_server_250_2**)

  そのレプリカがデフォルトのレプリカになり、そのレプリカが存在するセルがデフォルトのセルになります。
- レプリカが稼働しているホストのネットワーク・アドレス (たとえば、**ncadg_ip_udp:15.22.144.248**)

  そのホスト上のレプリカがデフォルトのレプリカになり、ホストが存在するセルがデフォルトのセルになります。

dcecp コマンドの一部は、マスター・レプリカに対してしか作動できないため、マスターへのバインディングが必要になります。マスターに対してのみ作動するコマンドを実行しており、そのマスターがデフォルトのレプリカでないときは、**dcecp** は、多くの場合、現行のデフォルトのセルの中のマスター・レプリカに自動的にバインドしようとします。その他の場合には、 **dcecp** はエラー・メッセージを表示し、コマンドは失敗します。

整合性のあるローカル・ファイルの保証

passwd_export コマンドは、ローカル・マシンの標準の **/etc/passwd** ファイルと **/etc/group** ファイルを、レジストリー・データベースと整合させます。 **passwd_export** >コマンドは定期的に (ただし、時期をずらせて)、できれば **cron** 処理の一部として実行します。 **passwd_export** が、新しいパスワードとグルー

プ・ファイルを正常に作成すると、現行のファイルを **passwd.bak** および **group.bak** という名前のバックアップとして保管します。作成が失敗すると、現行のファイルはそのまま残ります。

passwd_export コマンドの構文は次のとおりです。

```
passwd_export
[-n ][-d  directory_name] [-x ] [-m max_entries] [-s ] | [-h [elp] [-v ]
```

ここで、

-n **passwd_override** および **group_override** ファイル・エントリーを無視することを指定します。このフラグがないと、 **passwd_export** は、両方のファイルのオーバーライド・エントリーを、作成したローカル・パスワードとグループ・ファイルに適用します。

-d *directory_name*
passwd_export によって作成したローカル・パスワード・ファイルとグループ・ファイルを格納する、ディレクトリーの名前を指定します。このオプションを入力しなければ、デフォルトにより、ファイルはローカル・ノード上の **/etc** ディレクトリーに格納されます。

たとえば、**/etc/locals** というディレクトリーにファイルを格納する場合、次の形式でコマンドを入力します。

```
dceshared/bin/passwd_export -d/etc/locals
```

-x 暗号化されたパスワードとして **OMIT** を指定するパスワードまたはグループのオーバーライドを持つ (ローカル・マシン上で)、ユーザーのエントリーの作成を禁止します。 **passwd_export** によって作成されたパスワード・ファイルとグループ・ファイルから、省略されたユーザーまたはグループを除外するには、**-x** オプションを使用します。ユーザーを省略するためには、そのユーザーのオーバーライド・エントリーを作成して、ユーザーのパスワード・フィールド・エントリーとして **OMIT** という単語を入力しなければなりません。省略されたユーザーは、ローカル・マシンにログインできません。
(464ページの『ローカル・パスワード・ファイルからのユーザーの省略』を参照。) グループを省略するためには、そのグループのオーバーライド・エントリーを作成して、グループのパスワード・フィールド・エントリーとして **OMIT** という単語を入力します。

-m *max_entries*
/etc/passwd ファイルと **/etc/group** ファイルに入る、レジストリー・エントリーの最大数を設定します。

-s **/etc/passwd** ファイルと **/etc/group** ファイルのエントリーを、 UNIX 番号によってソートします。このオプションを指定しない場合、エントリーはレジストリーから検索された順にランダムに並びます。

-h[elp]
ヘルプ情報を表示します。

-v 冗長モードで実行します。

第37章 ネットワーク再構成の処理

この章では、レジストリー・レプリカの位置を変更する、ネットワーク再構成を処理するための手順について説明します。具体的には、この章では以下について説明します。

- マスター・レジストリー・サイトの変更
- ネットワークからのノードの削除
- ネットワーク・アドレス変更の処理

この章の手順を実行するには、管理アカウントをとおしてネットワーク・レジストリー・アカウントにログインする必要があります。

マスター・レプリカ・サイトの変更

マスター・レプリカ・サーバーを実行するマシンは、いつでも使用可能である必要があります。このマシンをネットワークから取り去るか、長期間シャットダウンさせることを計画している場合は、マスター・レプリカのサイトを変更する必要があります。

マスター・レジストリー・サイトを変更する方法としては、**dcecp registry designate** コマンドを使用して、マスター・サーバーとスレーブ・サーバーの役割を逆にすることをお勧めします。つまり、マスターをスレーブにして、スレーブをマスターにします。詳しくは、504ページの『現行のマスター・レプリカが失敗したときの新しいマスター・レプリカの指定』を参照してください。

dcecp registry designate コマンドを呼び出すと、以下のことが発生します。

1. 現行マスターが、すべての保留中の更新とその伝搬キューを、新規マスターとして指定されたレプリカに送信する。
2. 指定された新規マスターは、スレーブへの伝搬を管理するために必要な情報を入手するために、現行マスターのレプリカ・リストを読み取る。
3. 指定された新規マスターが、現行マスターから必要な情報をすべて入手すると、それが新規マスターになり、現行マスターがスレーブになる。

情報をこのように順序正しく完全に転送することによって、データが失われないことが保証されるので、マスター・サイトとスレーブ・サイトのレジストリー・サーバーが正常に作動している場合、マスター・レジストリーを別のマシンに移動する方法として、 **dcecp registry designate** コマンドをお勧めします。 **dcecp registry designate -master** コマンドは、レプリカをマスターからスレーブに変更するためにも使用できることに注意してください。しかし、 **dcecp registry designate -master** コマンドによってデータが失われる可能性があるため、現行マスターが破棄された場合にのみ このコマンドを使用してください。ネットワーク障害のため、あるいはマスターが一時的にダウンしたためにマスターが到達不能な場合には、これはお勧めしません。現行マスターが障害を起こすか、破棄されたとき、新規マスターのセキュリティー・レプリカを指定する方法の詳細については、このガイドの 504ページの『現行のマスター・レプリカが失敗したときの新しいマスター・レプリカの指定』を参照してください。

マスター・レプリカのサイトを変更する場合には、次のステップに従います。

1. 新しいマスター・サイトを選択する。スレーブ・レプリカがこのサイトに存在しなければなりません。必要であれば、**config.dce** コマンドを使用して、スレーブ・マシンを構成します。
2. Tcl **set** コマンドを実行して、デフォルトのレプリカを現行マスター・レプリカに設定する。次の例では、マスター・レプリカは、セル **giverny.com** の中の **oddball** という名前のレプリカに設定されます。

   ```
   dcecp> set _s(sec) /.../giverny.com/subsys/dce/sec/oddball
   dcecp>
   ```

3. **registry designate** コマンドを実行して、マスターとスレーブの役割を逆にします。このコマンドには、新しくマスターにするレプリカの名前を引き数として指定します。次の例では、 **/.../giverny.com/subsys/dce/sec/music** という名前のレプリカを新規マスターにします。

   ```
   dcecp> registry designate /.../giverny.com/subsys/dce/sec/music
   dcecp>
   ```

4. マスター・サイトが変更されたことを確認する。それは、**registry show -replica** コマンドを実行することにより行います。

ネットワークからのサーバー・マシンの除去

注:

1. DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。
2. 以下は、レプリカを削除するための手作業の指示です。削除したいレプリカが永続的に使用不可になったり、破棄されたときにのみ、これらの指示に従ってください。 DCE 3.2 の場合、レプリカ削除のためのお勧めする方法は、削除したいレプリカを持つホストに対し **unconfig.dce** コマンドを使用する方法です。レプリカを構成解除する場合は、 **unconfig.dce** を使って **sec_rep** コンポーネントを指定してください。

スレーブ・レプリカを実行するマシンをネットワークから撤去するとか、マシンを長期間シャットダウンすることを計画している場合、そのサイトのレプリカを削除します。

マスター・サーバーを実行しているノードを除去する場合、ノードを除去する前に、前述したようにマスター・サーバー・サイトを変更する必要があります。

スレーブ・レプリカを削除するためには、**dcecp registry delete** コマンドを使用します。このコマンドを実行すると、マスターは以下の処理を実行します。

1. レプリカを削除済みとマークする。
2. レプリカ・リスト上のすべてのレプリカに対して、削除を伝搬する。
3. 削除要求をレプリカに送達する。
4. レプリカ・リストからレプリカを削除する。

次のサンプル・コマンドは、 **/.../giverny.com/subsys/dce/sec/art_1** という名前のスレーブ・レプリカを削除します。

```
dcecp> registry delete /.../giverny.com/subsys/dce/sec/art_1
dcecp>
```

このコマンドを実行すると、**dcecp** は、必要ならば、現行セルにあるマスター・レプリカにバインドし、その後、マスター・レプリカはスレーブ・レプリカに対してスレーブ自身を削除するように指示します。

スレーブが削除されていることを確認するために、 **dcecp registry catalog** コマンドを実行します。マスターがスレーブを削除する要求を受け取っていると、スレーブは、削除用のマークが付いて、レプリカ・リストに表示されます。レプリカが実際に削除されると、リストに表示されなくなります。

ネットワーク・アドレス変更の処理

secd が始動すると、マスター・レプリカとスレーブ・レプリカはアドレス変更を検出することができ、マスターのレプリカ・リストとセル・ネームスペースに対して必要な更新を行うことができます。一般に、ネットワーク・アドレス変更を処理するためにユーザー側で必要なことは、 **pe_site** ファイルを更新することだけです。しかし、マスターとスレーブ・レプリカのネットワーク・アドレスが同時に変更されると、ユーザーの介入が必要になります。このサブセクションでは、 **pe_site** ファイルを更新する方法と、同時アドレス変更を処理する方法について説明します。

pe_site ファイルの更新

マスターまたはスレーブのネットワーク・アドレスを変更する場合には、セル内のすべてのマシンの **/opt/dcelocal/etc/security/pe_site** ファイルを更新する必要があります。このことは、特にセキュリティー・サーバーを稼働しているマシン上で、DCE を再始動する前に実行する必要があります (**secd**)。このファイルはセル内の各マシンに存在しており、 DCE セキュリティー・サービスがそれ自体にバインディングするためにこれが必要です。このファイルでは、セル内の各セキュリティー・サーバーで使用するインターフェースごとに、エントリーが示されています。セキュリティー・サーバー・マシンの場合、そのマシンのインターフェース情報が、ファイルの先頭に示されます。セキュリティー・サーバー・マシン以外の場合には、エントリーは、そのマシンで使用される ″lan-profile″ で指定した順番 (優先レプリカ順) か、ランダムな順番に並べられます。

(**root** ユーザーとして) 次のコマンドを使用し、 **pe-site** ファイルを更新します。

```
dcecp -c secval update
```

このコマンドが失敗した場合、 **/opt/dcelocal/etc/security/pe_site** を手動で更新します。

次に示すのは、2 つのセキュリティー・サーバー・マシンを指定した、 ″my_cell″ というセルからのサンプルの **pe_site** ファイルです。

```
/.../my_cell 5ddfd8c4-eb13-11d4-83a9-09356135aa77@ncacn_ip_tcp:9.53.96.52[]
/.../my_cell 5ddfd8c4-eb13-11d4-83a9-09356135aa77@ncadg_ip_udp:9.53.96.52[]
/.../my_cell 5ddfd8c4-eb13-11d4-83a9-09356135aa77@ncacn_ip_tcp:9.53.97.53[]
/.../my_cell 5ddfd8c4-eb13-11d4-83a9-09356135aa77@ncadg_ip_udp:9.53.97.53[]
```

同時アドレス変更の処理

マスター・レプリカとスレーブ・レプリカに対してアドレス変更が同時に行われると、マスターとスレーブの両方が互いに変更されたアドレスを通知しようとしている間、両方とも互いに到達することができません。この問題を避けるには、 1 つの

レプリカ (マスターまたはスレーブのいずれか) のアドレス変更をすべてのレプリカに伝搬してから、それ以外のアドレスを変更するようにします。 1 つのアドレス変更を行います。次に、**dcecp registry show -replica** コマンドを使用して、マスター・サイトとスレーブ・レプリカ・サイトの、両方のレプリカ・リストを表示します。両方のレプリカ・リストに新しいアドレスが表示されてから、次のネットワーク・アドレス変更を行うのが安全です。

マスターとスレーブに対する同時のネットワーク・アドレス変更を防ぐことができない場合、マスターとスレーブの間の通信をリストアする唯一の方法は、スレーブをいったん削除し、その後、それを再作成する方法です。ユーザーの状況に応じて、次のいずれかの方法でスレーブを削除します。

- マスターとスレーブがまだ通信を行っている間に、同時アドレス変更が予想される場合、 **set** コマンドを使用してマスターにバインドしてから、 **dcecp registry delete** コマンドを使用してスレーブ・レプリカを削除する。
- **secd** がマスター・サイトとスレーブ・サイトで作動中であるが、マスターとスレーブは通信をしていない場合、まず **set** コマンドを使用してスレーブにバインドし、次に **registry destroy** コマンドを使用してスレーブを破棄する。次に、**set** を使用してマスターにバインドするとともに、 **registry delete -force** コマンドを使用して、スレーブのレプリカ・リスト・エントリーを削除する。
- **secd** がスレーブ・サイトで作動中でないか、スレーブ・サイトにバインドできない場合には、 493ページの『第40章 トラブルシューティング手順』で説明されている、レプリカ再作成の手順を使用する。

第38章 レジストリーの設定

DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

この章では、既存の DCE セキュリティー・サービスのレジストリーを設定するために行う手順について説明します。これらの手順の一部は、DCE のインストールと構成の際に、 **dce_config** スクリプトによって自動的に処理されます。その他の手順は、DCE ユーティリティーと制御プログラムを使用して、ユーザーが行います。レジストリーを設定する手順は以下のとおりです。

1. ネットワーク内でセキュリティー・サービス・コンポーネントを配置する場所を計画する。
2. マスター・レジストリー・データベースを作成する (システム構成時に **dce_config** スクリプトにより行われる)。
3. マスター・レプリカを始動する (システム構成時に **dce_config** スクリプトにより行われる)。
4. レジストリー・データベースを移植する (ユーザーが **dcecp rgy_edit** コマンドを使用して行う)。
   - ポリシーと属性を設定する。
   - 名前とアカウントを追加する。
5. スレーブ・データベースを作成し、スレーブ・レプリカを始動する (システム構成時に **config.dce** スクリプトにより行われる)。
6. DCE がベースとなっているマシンすべてで **passwd_export** を実行するように、**cron** を設定して、ローカル・パスワード・ファイルとグループ・ファイルがレジストリーとの整合性を保つようにする (標準の UNIX コマンドを使用して、ユーザーが行う)。 **passwd_export** コマンドについては、 457ページの『第36章 ルーチン保守の実行』で説明されています。

レジストリーはセル・ディレクトリー・サービス (CDS) を使用してネットワーク・リソースについての情報を入手するため、この章では使用するネットワークが CDS 操作用に適切に構成されていることを前提としています。

DCE セキュリティー・サービス・コンポーネントのサイトの計画

ネットワークでセキュリティー・サービスを構成するためにまず行うことは、レジストリーのマスター・レプリカとスレーブ・レプリカ用のサイトを選択することです。これらのサイトでは、**secd**、セキュリティー・サーバーを実行します。 **secd** を実行するマシンは、常時作動しており、使用可能である必要があります。特に、マスター・レプリカを実行するマシンは、ネットワーク全体で使用可能であることが重要です。

secd の実行に必要なマシン・サイズは、プラットフォームとオペレーティング・システムによって異なります。ごく一般的な規則として、レジストリー・データベースの将来の拡張に十分対応できる容量のマシンを選択してください。マシンには、

レジストリー・データベース用の十分なディスク・スペースと、処理がスラッシング状態にならないように、十分なバッキング・ストアが必要です。

スクリプトを実行する際、スクリプトは、DCE ホスト・デーモン (**dced**) と必要な CDS サーバーを実行するためのマスター・レプリカ・サイトを構成します (DCE ホスト・デーモンは、ローカル・ホストにエンドポイント・マッパー・サービスを提供します)。

マスター・レジストリー・データベースの作成

セルのセキュリティー・サーバーを最初に構成するとき、 **config.dce** スクリプトは、 *dcelocal* **/bin/sec_create_db** コマンドを呼び出して、マスター・レプリカを作成します。 **sec_create_db** は、新規マスター・レプリカの作成時に、名前とアカウントを使って、そのデータベースを初期化します。 **sec_create_db** を実行するためには、ユーザーは **root** でなければならないことに注意してください。

sec_create_db コマンドは、マスター・レプリカのセル名とネットワーク・アドレスを含む、レジストリー構成ファイル (名前は *dcelocal***/etc/security/pe_site**) も作成します。このファイルは、CDS が使用できない場合、そのマシンで稼働しているクライアントに、 **secd** サーバーのバインド・アドレスを提供します。

新規のマスター・レジストリー・データベースを作成する必要が万一生じた場合には、 **sec_create_db** を直接起動することができます。 **sec_create_db** を実行するためには、ユーザーは root でなければならないことに注意してください。また、マスター・データベースを再作成することは非常にまれですが、スレーブが破棄された場合には、スレーブ・データベースの再作成が必要になる場合があることにも注意してください。以下のサブセクションでは、 **sec_create_db** コマンドの使用方法について説明します。

sec_create_db コマンドのフォーマット

sec_create_db コマンドのフォーマットは次のとおりです。

```
sec_create_db {-master | -slave} -my[name] my_server_name ¥
[-k[eyseed] keyseed] [-cr[eator]creator_name] ¥
[-cu[nix_id] creator_unix_id] [-u[uid] cell_uuid, ] ¥
[-p[erson_low_unix_id] unix_id]
[-g[roup_low_unix-id] unix_id] ¥
[-o[rg_low_unix-id] unix_id] [-ma[x_unix_id] unix_id] ¥
[-pa[ssword] default_password] [-ldap] [-v[erbose]
```

ここで、

-master
マスター・レプリカのデータベースを作成することを指定します。他の **sec_create_db** オプションはすべて、 **-master** オプションとともに使用することができます。

-slave スレーブ・レプリカのデータベースを作成することを指定します。 **-slave** オプションとともに使用できるのは、 **-myname**、**-keyseed**、および **-verbose** の各オプションのみです。

-my[name] *my_server_name*
これは、このマシンのセキュリティー・サーバー (**secd**) に割り当てる名前

です。この名前は、このセルのセキュリティー・サーバーを見付けるために、ネーム・サービスによって使用されます。

-k[eyseed] *keyseed*
これは、作成するデータベースのためのマスター・キーを作成するために、ランダム・キー生成プログラムに乱数の初期値を与えるストリングです。簡単に推測できないストリングにする必要があります。マスター・キーは、すべてのアカウント・パスワードの暗号化に使用されます。レプリカのそれぞれのインスタンス (マスターまたはスレーブ) は、自身のマスター・キーを持っています。 **dcecp registry modify** コマンド、または **sec_admin master_key** コマンドを使用して、マスター・キーを変更することができます。 (マスター・キーの変更のための **dcecp registry modify** コマンドの使用に関する詳細については、 457ページの『第36章 ルーチン保守の実行』を参照してください。) このオプションを入力しないと、 **sec_create_db** が入力するようプロンプトを出します。

-cr[eator] *creator_name*
これは、レジストリー作成者の名前です。レジストリー作成者は、レジストリー・データベースの初期の特権ユーザーです。 **dcecp acl modify** コマンド、または **acl_edit modify** コマンドを使用して、レジストリー・データベースの ACL を変更することによって、同じ特権をいつでも別のユーザーに与えることができることに注意してください。レジストリーの作成時に、レジストリー・オブジェクトのデフォルトの ACL エントリーも作成されます。このエントリーは、 **-cr** オプションに指定されているプリンシパルに対して、最高特権の許可を与えます。レジストリー作成者として指定されたプリンシパルが予約名のいずれでもない場合、 **sec_create_db** は、プリンシパルと、そのプリンシパルのアカウントを追加します。このオプションを入力しない場合、レジストリー・データベースの初期の特権ユーザーは **root** です。

-cu[nix_id] *creator_unix_id*
これは、レジストリー作成者に割り当てるために指定する UNIX 番号です。このオプションを入力しない場合、レジストリー作成者の UNIX 番号は動的に割り当てられます。

-u[uid] *cell_uuid*
これはセルの UUID です。この UUID を入力しない場合、UUID は動的に割り当てられます。

-p[erson_low_unix_id] *unix_id*
これは、**dcecp registry modify** コマンド、または **rgy_edit properties** コマンドを使用してプリンシパルが追加されるときに自動的に生成される、UNIX ID の開始点です。この番号よりも小さい UNIX ID を明示的に入力できることに注意してください。この下限は、自動的に生成される UNIX ID 番号のみに適用されます。

-g[roup_low_unix_id] *unix_id*
これは、**dcecp registry modify** コマンド、または **rgy_edit properties** コマンドを使用してグループが追加されるときに自動的に生成される、UNIX ID の開始点です。この番号よりも小さい UNIX ID を明示的に入力できることに注意してください。この下限は、自動的に生成される UNIX ID 番号のみに適用されます。

-o[rg_low_unix_id] *unix_id*

これは、**dcecp registry modify** コマンド、または **rgy_edit properties** コマンドを使用してオーガニゼーションが追加されるときに自動的に生成される、 UNIX ID の開始点です。この番号よりも小さい UNIX ID を明示的に入力できることに注意してください。この下限は、自動的に生成される UNIX ID 番号のみに適用されます。

-ma[x_unix_id] *unix_id*

これは、プリンシパル、グループ、またはオーガニゼーションが追加されるときに、 UNIX ID として割り当てることができる最大の数値です。この数値より大きい UNIX ID が自動的に割り当てられることはなく、ユーザーがこの数値より大きい数値を明示的に入力することもできません。最大 UNIX ID は、**dcecp registry modify** コマンド、または **rgy_edit properties** コマンドを用いて変更するまで、そのまま残ります。

-pa[ssword] *default_password*

これは、**sec_create_db** によって作成されるアカウントに割り当てられる、デフォルトのパスワードです。デフォルトのパスワードを指定しない場合、 **-dce-** が使用されます。アカウント **hosts/***local_host_name/principal_name***none none**、 **krbtgt/***cell_name* **none**、および **nobody none none** にはデフォルトのパスワードは割り当てられず、代わりに、ランダムに生成されたパスワードが割り当てられることに注意してください。

-ldap セキュリティー・レジストリーを LDAP データベースに保管することを示します。

-v[erbose]

冗長モードで実行し、すべての活動の冗長なトランスクリプトを生成します。

sec_create_db の実行例

次の例は、マスター・データベースを作成するために実行する **sec_create_db** コマンドと、 **sec_create_db** が実行時に表示する情報を示しています。 **-k** オプションが入力されないので、 **sec_create_db** がマスター・キーの乱数の初期値のストリングを入力するようプロンプト指示することに注意してください。このストリングは、入力時には表示されません。

```
/work/krb/sec_create_db -v -myname /.../dresden.com/subsys/dce/sec/master
-master
Enter keyseed for initial database master key: <enter up to
1024 characters>
SECD Checkpoint on Tue Sep 27 11:44:12 1994
....  saving rgy
....  saving acct
....  saving person
....  saving group
....  saving org
....  saving replicas
....  saving acl
End SECD Checkpoint on Tue Sep 27 11:44:13 1994
SECD Checkpoint on Tue Sep 27 11:44:15 1994
....  saving rgy
....  saving acct
....  saving person
....  saving group
```

```
....  saving org
....  saving acl
End SECD Checkpoint on Tue Sep 27 11:44:17 1994
```

sec_create_db の結果

sec_create_db によって作成されるマスター・レジストリー・データベースには、以下の表にリストされたプリンシパル、グループ、およびオーガニゼーションが含まれています。

表 26. 初期のプリンシパル、グループ、およびオーガニゼーション

プリンシパル	グループ	オーガニゼーション
bin	**bin**	none
daemon	**daemon**	—
dce-ptgt	**kmem**	—
dce-rgy	**mail**	—
krbtgt/ *local_cell_name*	**nogroup**	—
hosts/ *local_host*/**self**	**none**	—
mail	**system**	—
nobody	**tcb**	—
root	**tty**	—
sys	**uucp**	—
tcb	—	—
uucp	—	—
who	—	—

sec_create_db コマンドが作成するアカウントは次のとおりです。

- **bin bin none**
- **daemon daemon none**
- **dce-ptgt none none**
- **dce-rgy none none**
- **hosts/***local_host*/**self none none**
- **krbtgt/** *cell_name* **none none**
- **nobody nogroup none**
- **root system none**
- **uucp uucp none**

sec_create_db によって初期に作成されたオブジェクトの一部は、予約済みであり、削除することができません。これらのオブジェクトを以下のリストに示します。

- 予約済みのプリンシパルは次のとおりです。
  - **dce-ptgt**
  - **krbtgt/** *cell_name*
  - **dce-rgy**
- 予約済みアカウントは、以下のとおりです。
  - **dce-ptgt none none**
  - **krbtgt/** *cell_name* **none none**
  - **dce-rgy none none**

sec_create_db コマンドを実行して、マスター・レジストリー・データベースを作成するとき、レジストリーへの最高特権のアクセスを持つプリンシパルを指名できます。 この人はレジストリー作成者と呼ばれます。 指名するレジストリー作成者がデフォルト・プリンシパルのいずれでもない場合、 **sec_create_db** は、アカウント *rgy_creator* **none none** を追加します。ここで、*rgy_creator* はレジストリー作成者として指名したプリンシパルです。レジストリー作成者を指名しないと、**sec_create_db** は最高特権のレジストリー・アクセスを **root system none** アカウントに割り当てます。

1 つの例外を除き、 **sec_create_db** コマンドによって作成されたすべてのアカウントには、ランダムに生成されたパスワードが割り当てられ、無効であるというマークが付けられます。これらのプリンシパルがこれらのアカウントにログインできるようするには、その前に、アカウント・パスワードを変更して、アカウントに有効であるというマークを付けなければなりません。 **dcecp account modify** コマンドを使用して、これを行うことができます。 395ページの『第31章 アカウントの作成および保守』には、レジストリー内のプリンシパルのアカウントの属性のすべて (プリンシパルのパスワードを含む) を変更するための **dcecp account modify** コマンドの使用法に関する指示があります。また、**dcecp** には、新規パスワードをランダムに生成するためのオプションもあります。

ただし、例外は、レジストリー作成者に対して作成されたアカウントが有効であり、 DCE デフォルト・パスワード (**-dce-**) が割り当てられます。レジストリー作成者のアカウントのセキュリティーを保証するために、デフォルトのパスワードを変更してください。

sec_create_db によって作成されたアカウントにより暗黙的に指定されるグループ・メンバーシップに加え、プリンシパルは、表27 にリストするグループのメンバーにもなります。

表 27. sec_create_db によって作成されるグループ・メンバーシップ

このプリンシパルは...	このグループのメンバーである...
who	**bin**
root	**system**
	kmen
	tty
sys	**kmem**
mail	**mail**
tcb	**tcb**

357ページの『第30章 プリンシパル、グループ、および組織の作成および保守』には、プリンシパルをグループに追加するための指示があります。

マスター・レプリカの始動

dce_config は、マスター・レプリカを作成した後、マスター・レプリカを始動します。マスター・レプリカ (**secd**) を明示的に始動するためには、次の手順を実行します。

1. マスター・レプリカを実行するマシンに **root** としてログインする。

2. **start.dce** を実行します。

新規レジストリー・データベースの移植

マスター・レプリカを作成し、始動したら、ポリシーとプロシージャーを設定し、そしてアカウントを追加することによって、データベースを移植する必要があります。

ポリシーと属性の設定

ポリシーと属性を表示して、必要に応じて変更するためには、 **dcecp registry show** コマンドと **dcecp registry modify** コマンドを使用します。

アカウントの追加

新規レジストリー・データベースの作成後、そこに含まれているものは、**sec_create_db** により初期情報として追加された、プリンシパル、グループ、オーガニゼーション、およびアカウントだけです。サイトが必要とするその他の名前とアカウントを追加するために、 **dcecp account create** コマンドを使用します。これは今すぐにでも、あるいはその後でいつでも行うことができます。 **dcecp** を使用してのアカウントの追加についての詳細は、 395ページの『第31章 アカウントの作成および保守』 を参照してください。

スレーブ・レプリカの作成

マスター・レプリカ・データベースを作成、始動し、そのデータベースの移植後、スレーブ・レプリカを作成し、始動するために、スレーブ・サイトで **dce_config** を実行します。スレーブ・レプリカを作成、始動するために、**dce_config** は、まずサイトが **dced** と適切な CDS サーバーを実行中であることを確認します。次に、以下の **sec_create_db** コマンドを実行します。

```
dcelocal/bin/sec_create_db -slave
-myname my_server_name
```

まず、コマンドは新規のスレーブ・レプリカ用のデータベースを作成します。このデータベースはスタブ・ファイルのみで構成されています。次にコマンドは、マスター・レプリカを見付けて、マスターのレプリカ・リストに新規のスレーブを追加します。マスターは初期化のために、新規のレプリカにマークを付けます。最後に、 **dce_config** スクリプトが **secd** を始動し、マシンのリブートごとに自動的に始動することを確認します。

スレーブ・レプリカを実行させたいそれぞれのマシンに、スレーブ・レプリカを構成するために、 **dce_config** を実行する必要があります。

レプリカの実行の確認

マスター・レプリカとスレーブ・レプリカを正しい場所に置いて始動したあと、レプリカが実行されていることを確認するために、次の手順を行います。

1. セルで稼働しているセキュリティー・サーバーの名前のリストを入手する。

```
dcecp> registry catalog
/.../dc.cell.ch.hp.com/subsys/dce/sec/dce6
/.../dc.cell.ch.hp.com/subsys/dce/sec/dce5
dcecp>
```

2. レプリカが保持している伝搬情報を見る。

```
dcecp> registry show /.../dc.cell.ch.hp.com/subsys/dce/sec/dce6
-replica
{name /.../dc.cell.ch.hp.com/subsys/dce/sec/dce6}
{type slave}
{cell /.../dc.cell.ch.hp.com}
{uuid 07f5c1dc-80ef-11cf-b60c-0800095f6636}
{status enabled}
{lastupdtime 1996-09-17-13:06:53.000-04:00I-----}
{lastupdseq 0.33826}
{addresses
 {ncadg_ip_udp 15.22.51.49}
 {ncadn_ip_tcp 15.22.51.49}}
{masteraddrs
 {ncadg_ip_udp 15.22.48.183}
 {ncadn_ip_tcp 15.22.48.183}}
{masterseqnum 0.15724}
{masteruuid 90067612-d3f6-11ce-b773-0800095a49d8}
{supportedversion
 secd.dce.1.0.2
 secd.dce.1.1}
dcecp>
```

3. マスターが各スレーブごとに保持している、伝搬情報を見る。

```
dcecp> registry show /.../dc.cell.ch.hp.com/subsys/dce/sec/dce5
-master
{name /.../dc.cell.ch.hp.com/subsys/dce/sec/dce6}
{uuid 07f5c1dc-80ef-11cf-b60c-0800095f6636}
{type slave}
{addresses
 {ncadg_ip_udp 15.22.51.49}
 {ncadn_ip_tcp 15.22.51.49}}
{propstatus update}
{lastupdtime 1996-09-17-13:06:53.000-04:00I-----}
{lastupdseq 0.33826}
{numupdtogo 0}
{commstate ok}
{lastcommstatus {Successful completion}}
{name /.../dc.cell.ch.hp.com/subsys/dce/sec/dce5}
{uuid 90067612-d3f6-11ce-b773-0800095a49d8}
{type master}
{addresses
 {ncadg_ip_udp 15.22.48.183}
 {ncadn_ip_tcp 15.22.48.183}}
dcecp>
```

第39章 DCE への UNIX アカウントのインポート

passwd_import コマンドは、 **/etc/passwd** ファイルと **/etc/group** ファイルの情報に基づいたエントリーを、レジストリー内に作成します。このコマンドを使用すれば、 DCE セキュリティー・サービスを使用するマシンと使用しないマシンとの間のアカウントの整合性が保証され、既存の UNIX ユーザー・ベースをレジストリーに追加できます。

passwd_import の動作

passwd_import は、エントリーを処理するときに、グループ・ファイルとパスワード・ファイルのエントリーをレジストリー・エントリーと比較します。以下の 2 種類の競合を検出することができます。

名前の競合

この競合は、レジストリーと、グループ・ファイルまたはパスワード・ファイルに同じ名前のストリングが定義されている場合に起こります。名前 **joe 102** と **joe 555** は、このような競合の例です。重複した名前が、同じユーザーを表すことも、2 人の異なるユーザーを表すこともあります。

UNIX ID 競合

この競合は、名前の異なるユーザーのレジストリーと、グループ・ファイルまたはパスワード・ファイルに同じ UNIX ID が定義されている場合に起こります。 **joe 102** と **ann 102** は、このような競合の例です。

これらの競合は、上記の例のように、別々に検出されたり、一緒に検出されることがあります。たとえば、レジストリー・エントリー **joe 102** と、 UNIX エントリー **joe 102** は競合しています。競合が検出された場合、パスワード・ファイルとグループ・ファイルのエントリーの変更のために使用する情報を提供するか、あるいは **passwd_import** にそのエントリーをインポートしないように指示する必要があります。 **passwd_import** コマンドは、既存のレジストリーのプリンシパル、グループ、またはアカウントには変更を加えませんが、ユーザーが指定すれば、グループ・ファイルとパスワード・ファイルに基づいた、新規のプリンシパル、グループ、およびアカウントをレジストリーに作成します。

passwd_import の処理ステップ

passwd_import は、エントリーを処理するときに、以下のステップを順に実行します。

1. グループ・ファイルとパスワード・ファイルをオープンし、レジストリーへの接続を確立する。
2. グループ・ファイルのエントリーをレジストリーのグループと比較する。競合がない場合は、グループ・ファイル内のグループに対応するグループを、レジストリー内に作成する。
3. パスワード・ファイルのエントリーをレジストリーのプリンシパルと比較する。ここでも、競合がなければ、次のことを行う。
   - パスワード・ファイル内のエントリーに対応するプリンシパルを、レジストリー内に作成する。

- 新しく作成したプリンシパルを該当するグループに追加する。
- 新しく作成したプリンシパルのアカウントを作成する。

4. グループ・ファイルを再検査して、そこで検出した追加のグループのメンバーとしてプリンシパルをそのグループに追加する。

レジストリーに対する変更は、それぞれのステップの処理時に個別に行われます。**-o** オプションを指定すると、**passwd_import** は、新しく作成したレジストリー・プリンシパルをすべて、指定されたオーガニゼーションに追加します。ユーザーがオーガニゼーションを指定していない場合、プリンシパルは、オーガニゼーション **none** に追加されます。

passwd_import によって作成されるレジストリー・エントリー

エントリーがパスワード・ファイルまたはグループ・ファイルに存在し、レジストリーには存在しない場合、 **passwd_import** は新規のレジストリー・エントリーを作成します。追加のレジストリー情報に対して、 **passwd_import** は以下の値を取ります。

- プリンシパル・エントリーとグループ・エントリーの場合
  - **別名 /1 次名** = パスワード・ファイルに同一の UNIX 番号のエントリーが 2 つある場合、 **passwd_import** は検出した最初の UNIX 番号に対し 1 次名エントリーを作成し、同じ UNIX 番号が生じたときにはそれぞれに対して別名を作成する。
  - **フルネーム** = ブランク・ストリング。エントリーにフルネームは追加されない。
  - **メンバーシップ・リスト** = 新規グループの場合のみ、グループ・ファイルにリストされたすべてのプリンシパルと、そのグループに関連付けられたレジストリー・アカウントを持つすべてのプリンシパル。
  - **プロジェクト・リスト** = Yes (グループの場合のみ)。
- アカウント・エントリーの場合
  - **アカウント有効期限** = None。
  - **アカウント有効フラグ** = No。パスワード設定後に、このフラグを **y** に変更するために、 **dcecp account modify** コマンドを使用します。
  - **クライアント・フラグ** = Yes。
  - **重複証明フラグ** = No。
  - **転送可能証明フラグ** = Yes。
  - **GECOS** = **etc/passwd** ファイルにある、プリンシパルの GECOS フィールドのエントリーと同じ値。
  - **発効日** = アカウントの作成時刻。
  - **ホーム・ディレクトリー** = **/etc/passwd** ファイルにある、プリンシパルのホーム・ディレクトリーのエントリーと同じ値。
  - **ログイン・シェル** = **/etc/passwd** ファイルにある、プリンシパルのログイン・シェルのエントリーと同じ値。
  - **最大証明存続時間** = レジストリーの認証ポリシーに設定される。
  - **最大証明更新可能** = レジストリーの認証ポリシーに設定される。
  - **パスワード** = ランダムに生成される。ユーザー認証が可能になる前に、ランダムに生成されたパスワードを変更またはリセットする必要があることに注意してください。
  - **パスワード変更日時** = **passwd_import** が実行された日時に設定される。

- **パスワード有効フラグ** = No。
- **事後日付証明フラグ** = No。
- **代理可能証明フラグ** = No。
- **更新可能証明フラグ** = Yes。
- **サーバー・フラグ** = Yes。
- **TGT 認証フラグ** = Yes。

passwd_import は、作成するアカウントに対しては、使用可能なパスワードを設定しないことに注意してください。認証を可能にするには、その前に **dcecp account modify** コマンドを使用して、パスワードを設定する必要があります。また、アカウントが有効であることを確認する必要もあります。これは、次の例のように、 dcep account modify コマンドの -acctvalid オプションを指定することにより、新規パスワードを設定するのと同時に行うことができます。

```
dcecp> account modify John_Hunter -acctvalid {yes}
```

これに失敗すると、プリンシパルが **dce_login** を使用できなくなります。

passwd_import コマンドの構文

passwd_import コマンドの構文は次のとおりです。

```
dceshared/bin/passwd_import [-h] [-c] -d pathname [-i] [-o org] ¥
     [-p  password] [-u  username] [-v]
```

ここで、

-h 使用情報を表示します。

-c 検査モードで実行します。競合を表示するコマンドを処理しますが、レジストリーには変更を加えません。

-d *pathname*
インポートするパスワード・ファイルとグループ・ファイルを含んだ、ディレクトリーへのパス。

-i 同じ名前のストリングが競合せず、同じ識別を表すことを指定します。

-o *org* レジストリーに追加する、すべてのプリンシパルに割り当てるオーガニゼーションの名前。デフォルトは **none** という名前のオーガニゼーションです。

-p *password*
passwd_import の実行に使用する特権を持つアカウントのパスワード。 **-i** オプションを使用しないと、 **passwd_import** は、名前の競合を解決するようにプロンプトを出してきます。

-u *username*
passwd_import の実行に使用する特権を持つアカウントのプリンシパル名。このアカウントは、レジストリーにアクセスし、プリンシパル、グループ、アカウントとオーガニゼーション、およびメンバーをグループとオーガニゼーションに追加するための特権を持っている必要があります。プリンシパル名とパスワードは、ネットワーク認証を取得するために使用されます。これらを指定しないと、すでにネットワーク・ログインが行われている場合でも、 **passwd_import** は、その指定をプロンプトを出してきます。

-v 冗長モードで実行し、すべての活動の冗長なトランスクリプトを生成します。

passwd_import の使用

passwd_import を使用するためには、セキュリティー・サーバーが実行中である必要があります。以下のサブセクションでは、 **passwd_import** コマンドとそのオプションの使用方法について説明します。

同一ユーザー・オプションの使用

-i オプションを使用すれば、重複する名前が競合せず、同じ識別を表すことを指定することができます。 **passwd_import** は、重複する名前のエントリーを検出すると、そのエントリーを同じユーザーであるかのように処理し、それから次のエントリーにスキップします。

検査モードの使用

まず **-c** オプションを使用して、 **passwd_import** を検査モードで実行します。このモードでは、 **passwd_import** は処理の実行結果をシミュレートして、 **-c** オプションを指定せずに **passwd_import** を実行した場合に発生する競合を表示しようとします。

検査モードを使用すると、潜在的な競合の量と複雑さがよく分かるようになります。ただし、検査モードは、レジストリーに対しては変更を加えません。 **-c** オプションを指定せずに **passwd_import** を実行して、競合を解決するために変更を行った場合、今度はこの変更により、検査モードでは容易に明らかにならない、競合が発生することがあります。

検査モードで多数の競合が発生した場合は、一部の顕著な競合の解決のためには、**passwd_import** を実行する前に、レジストリーまたは UNIX グループ・ファイルとパスワード・ファイルのいずれかを手作業で編集した方がより効率的です。

競合の解決

passwd_import コマンドは、検出した競合を解決する方法の指示をプロンプトを出してきます。以下の選択肢があります。

- 別名を作成して UNIX ID の競合を解決する。このアクションは、競合しているレジストリー・オブジェクトに対して別名を作成します。この別名には、グループ・ファイルまたはパスワード・ファイル内の競合しているエントリーと同じ名前が割り当てられます。たとえば、エントリー **joe 555** がレジストリーに存在し、エントリー **tim 555** がパスワード・ファイルに存在する場合、このオプションは、**joe 555** に対して別名 **tim** を作成します。
- 新しい UNIX ID を自動的に生成するか、新しい UNIX ID を明示的に入力して、UNIX ID の競合を解決することができます。たとえば、レジストリーのエントリー **joe 555** とパスワード・ファイルの **tim 555** との間に競合がある場合は、 **tim** に対して新しい UNIX ID を生成することができます。
- 新しい名前を入力して、名前の競合を解決することができます。たとえば、レジストリーのエントリー **joe 555** とパスワード・ファイルの **joe 383** との間に競

合がある場合、 **joe 383** に対して新しい名前を生成することができます。この新しい名前はレジストリーに追加されます。

その他に、競合を無視して、エントリーをスキップするオプションもあります。

プロンプトへの応答

passwd_import を実行するときに、名前と番号 (UNIX ID) を入力するようプロンプト指示されることがあります。名前には、@ (アットマーク) と : (コロン) 以外の任意の文字または数字を含めることができ、長さは 1024 文字を超えてはなりません。

名前または番号を間違ったフォーマットで入力すると、 **passwd_import** は入力を無視し、再度プロンプトしてきます。

passwd_import セッションの例

このセクションでは、単純化した **passwd_import** セッションを示します。このサンプル・セッションは、以下に示すレジストリーのグループとパスワードのエントリーと、 UNIX のグループ・ファイルおよびパスワード・ファイルのエントリーを使用します。便宜上、レジストリー・エントリーは、パスワード・ファイルとグループ・ファイルの形式で示されています。ただし、レジストリー・エントリーは、レジストリー・データベース内にこのように格納されているわけではありません。

レジストリー・グループとパスワード・エントリー

- グループ・エントリー

```
wheel::0:
daemon::1:
none::2:
backup::3:user
locksmith::4:
login::5:
mail::6:bin
bin::7:root
server::8:
sys::9:root
staff::10:
sys_admin::11:user
sys_proj::12:
tgroup::35:
```

- パスワード・エントリー

```
root:sq1RclUrrb1L6:0:10::/:
daemon:sq1RclUrrb1L6:1:2::/:
none:sq1RclUrrb1L6:2:2::/:
user:sq1RclUrrb1L6:3:2::/:
lp:sq1RclUrrb1L6:4:7::/:
sys_person:sq1RclUrrb1L6:5:2::/:
admin:sq1RclUrrb1L6:6:2::/:
uucp:sq1RclUrrb1L6:7:2::/usr/spool/uucppublic:
bin:sq1RclUrrb1L6:8:7::/:
```

UNIX グループ・ファイルとパスワード・ファイルのエントリー

- グループ・ファイル・エントリー

```
system::0:root
other::1:
bin::2:root,bin,daemon
sys::3:root,bin,sys,adm
adm::4:root,adm,daemon
mail::6:root
rje::8:rje,shqer
daemon::12:root,daemon
tgroup::35:
diags::48:brown,smith,jones
cheetah::50:root,daemon
mkt_dev::52:roberts,anderson,hill
```

- パスワード・ファイル・エントリー

```
root::0:1:0000-Admin(0000):/:
daemon::1:1:0000-Admin(0000):/:
bin::2:2:0000-Admin(0000):/bin:
sys::3:3:0000-Admin(0000):/usr/src:
adm::4:4:0000-Admin(0000):/usr/adm:
uucp::5:5:0000-uucp(0000):/usr/lib/uucp:
rje::18:18:0000-rje(0000):/usr/rje:
trouble::70:1:trouble(0000):/usr/lib/trouble:
lp::71:2:0000-lp(0000):/usr/spool/lp:
setup::0:0:general system administration:/usr/admin:/bin/rsh
powerdown::0:0:general system administration:/usr/admin:/bin/rsh
sysadm::0:0:general system administration:/usr/admin:/bin/rsh
checkfsys::0:0:check diskette file system:/usr/admin:/bin/rsh
makefsys::0:0:make diskette file system:/usr/admin:/bin/rsh
mountfsys::0:0:mount diskette file system:/usr/admin:/bin/rsh
umountfsys::0:0:unmount diskette file system:/usr/admin:/bin/rsh
```

passwd_import の呼び出し

サンプル・セッションでは、 **passwd_import** コマンドがシェル・プロンプトで入力されます。

```
passwd_import -d sys5.3_tapes/adm -i
-v -u cell_admin
```

このコマンドは、以下のことを指定します。

- 同一の名前は同じ識別を表す (**-i**)。
- UNIX グループ・ファイルとパスワード・ファイルは、 **sys5.3_tapes/adm** ディレクトリーにある。
- コマンドは検査モードでは実行されない (**-c** が指定されていない)。
- コマンドは冗長モードで実行される (**-v** が指定されている)。
- 認証に使用するアカウントを持つプリンシパルは **cell_admin** である。
- **-p** オプションが指定されていないので、コマンドは **cell_admin** アカウントのパスワードを入力するようプロンプト指示する。

コマンドを呼び出すと、システムは以下のものを表示して、 **passwd_import** 処理の準備をします。

```
Preparing import files. (dce / sad)
Setting up registry information. (dce / sad)
Verifying that the necessary Organization exists. (dce / sad)
Creating group objects from group file. (dce / sad)
```

passwd_import は、UNIX のグループ・ファイルとパスワード・ファイルを読み取る際に、競合があればそれを通知して、競合を解決するようプロンプト指示します。

グループ・ファイルの検査

passwd_import コマンドは、まず名前の競合、次に UNIX ID の競合がないかどうか、グループ・ファイルを検査します。プロンプトに応答して競合を解決するときに、**passwd_import** は、指示があれば、レジストリー内にグループを作成します。

以下のステップは、グループの UNIX ID の競合がどのように処理されるかを示しています。

1. **passwd_import** コマンドはまず、前のセクションで示されているような、UNIX ID 間の競合を検出します。グループ・ファイル内の名前 **wheel** とレジストリー内の名前 **system** は、両方とも UNIX ID **0** を持っています。**passwd_import** コマンドは、次のように、競合を解決する方法の指示をプロンプトを出してきます。

   ```
   CONFLICT: (wheel 0) - Import Group's UNIX id exists in registry.
             (dce / sad)
             (system 0) is the conflicting entry from the registry.
   Do you wish to resolve the conflict (y) or skip this entry (n):
   ```

2. このエントリーをスキップするために **n** を入力すると、 **passwd_import** は処理を継続します。競合を解決するために **y** を入力すると、 **passwd_import** は競合を解決する方法の指示をプロンプトを出してきます。次の例では、**system** エントリーに対して別名 **wheel** をレジストリーに作成することによって、この競合が解決されます。

   ```
   Do you wish to resolve the conflict (y) or skip this entry (n): y
   Select one of: (a)lias, (g)enerate, (e)nter, (s)kip entry, (h)elp: a
   >> Adding Group entry for: wheel 0
   ```

   passwd_import は、冗長モードで実行中であるため、実行しているアクションを記述します。それぞれのアクション記述には、先頭に **>>** (リダイレクト記号) がつきます。

   検査モードで **passwd_import** を実行している場合は、競合を解決するようにプロンプト指示されることはありません。その代わりに、競合していることが知らされて、処理は継続します。このメッセージは次のように表示されます。

   ```
   CONFLICT: (wheel 0) - Import Group's UNIX id exists in registry.
           (dce / sad)
           (system 0) is the conflicting entry from the registry.
   Would need new UNIX id to resolve conflict. (dce / sad)
   ```

3. **passwd_import** は、解決する必要がある競合を検出しなかった場合、現在処理しているグループ・エントリーと、実行中のアクション (冗長モードで実行しているため) を表示します。次の例では、 **passwd_import** が **-i** オプションを指定して呼び出されたため、名前の競合を解決するためのプロンプトは出されません。

   ```
   CONFLICT: (tgroup 35)
    - Group name exists in registry and UNIX ids match.
      (dce / sad)
   >> Import Group:  - Ignoring name conflict, as instructed
   ```

```
   (dce / sad)
>> Adding Group entry for: diags 48
>> Adding Group entry for: cheetah 50
>> Adding Group entry for: mkt_dev 52
```

passwd_import は、UNIX グループ・ファイル全体の処理を継続する間に、 2 つの別の UNIX ID の競合を検出します。それは UNIX エントリー **adm 4** と **rje 8** で、それぞれ、レジストリー・エントリーの **locksmith 4** および **server 8** と競合します。

パスワード・ファイルの検査

passwd_import コマンドは、次に、パスワード・ファイルに競合がないかどうかの検査に進みます。検査が始まると、以下が表示されます。

```
Creating principal entries and accounts from password file.
(dce / sad)
```

競合なしにエントリーが処理されるとき、**passwd_import** は、レジストリーにプリンシパルを作成し、該当するグループとオーガニゼーションにプリンシパルを追加し、プリンシパルに対してアカウントを作成します。この時点で、以下が表示されます。

```
>> Adding Principal entry for: rje
>> Adding account for rje none.
```

次の例は、**passwd_import** が競合を検出したときに表示される、警告メッセージを示しています。

```
CONFLICT: (bin 2)
 - Principal name exists in registry and UNIX ids match.
   (dce / sad)
>> Import Principal:  - Ignoring name conflict, as instructed
       (dce / sad)
```

このメッセージは、 **bin** のアカウントがレジストリーに存在することを知らせるものです。両方のアカウントは、UNIX パスワード・ファイル・エントリーが、レジストリー・アカウントとは異なる情報を含んでいても、変更されないで残ります。

サンプル・セッションで使用しているコマンドで **-i** オプションが指定されているため、名前の競合は無視されます。以下の例は、 **-i** オプションを指定せずに **passwd_import** を実行した場合に検出される、名前の競合についてのプロンプトを示しています。

```
CONFLICT: (daemon 1)
 - Principal name exists in registry and UNIX ids match.
   (dce / sad)
Do you wish to resolve the conflict (y) or skip this entry (n):
```

n を入力すると、エントリーはスキップされ、処理が継続します。 **y** を入力すると、**passwd_import** は次のように、外部プリンシパルの新しい名前を入力するようにプロンプト指示します。

```
Enter new name for principal "daemon" "1":
```

グループへのメンバーの追加

passwd_import は、UNIX パスワード・ファイルの処理を完了すると、グループ・ファイルを再検査し、追加のグループを検出した場合、そのグループに新規作成のプリンシパルを追加します。このとき、以下が表示されます。

```
Add memberships from imported group file.  (dce / sad)
>> Add root as member of group with UNIX id: 0
>> Add root as member of group with UNIX id: 2
>> Add daemon as member of group with UNIX id: 2
```

処理の完了

passwd_import が処理を完了すると、以下が表示されます。

```
Closing import files.  (dce / sad)
Closing connection to registry.  (dce / sad)
```

第40章 トラブルシューティング手順

この章では、セキュリティー・サーバーのトラブルシューティングのための手順を説明します。これらの手順を使用するのは、ネットワークまたはハードウェアの障害によりレジストリーのオペレーションが中断された場合、または他の方法では解決できない問題が発生した場合に限られます。これらの手順は、以下のことを行う方法を示します。

- dce daemon のコア・ロケーションと別のファイルシステムへの symlink 方法のマッピング
- セキュリティー・サーバーの再始動
- セキュリティー・サーバーの障害後のシステムのバックアップと回復
- ロック・モードでのセキュリティー・サーバーの再始動
- レプリカのバックアップからのリストア
- スレーブ・レプリカの強制的削除
- ユーザー間プロトコルで登録済みのサーバーへのアクセス
- 所有者の削除のためオーファンになったレジストリー・オブジェクトの採用

手順を実行する前に、管理アカウントをとおしてログインしていることを確認する必要があります。

DCE Daemon のコア・ロケーションと別のファイルシステムへの Symlink 方法のマッピング

以下に daemons とそのコア・ダンプの場所のマッピングを示します。

- secd - /opt/dcelocal/var/security/adm/secd
- dced - /opt/dcelocal/var/dced
- cdsd - /opt/dcelocal/var/directory/cds/adm/cdsd
- gdad - /opt/dcelocal/var/directory/cds/adm/gdad
- csdadv - /opt/dcelocal/var/adm/directory/cds/cdsadv
- cdsclerk - /opt/dcelocal/var/adm/directory/cds/cdsclerk
- dtsd - /opt/dcelocal/var/time/adm/dtsd
- pwd_strengthd - /opt/dcelocal/var/security/adm/pwd_strength
- idmsd - /opt/dcelocal/var/security/adm/idmsd
- emsd - /opt/dcelocal/var/sysmgmt/adm/emsd
- dceagtd - /opt/dcelocal/var/sysmgmt/adm/dceagtd
- audit - /opt/dcelocal/var/audit/adm
- nsid - /opt/dcelocal/var/directory/cds/adm/nsid
- session_manager - /tmp/sessmgr
- session_manager_control - /tmp/session_control

以下に示すのは、AIX 固有の情報です。

- dceunixd - /opt/dcelocal/var/security/adm/dceunixd

以下に示すのは、Solaris 固有の情報です。

- nss_dce_server - /opt/dcelocal/var/security/adm/nss

以下のスキーム、またはこれに似たスキームを使用するようお勧めします。

SMIT (SMIT は AIX の場合のみ) を使用して、 /cores/dce と呼ばれる新規ファイル・システムを作成します。 /cores/dce が上記のディレクトリーのそれぞれに対して個別のディレクトリーを含むように、以下を発行します。

- mkdir /cores/dce/secd
- mkdir /cores/dce/dced
- mkdir /cores/dce/cdsd
- mkdir /cores/dce/gdad
- mkdir /cores/dce/cdsadv
- mkdir /cores/dce/cdsclerk
- mkdir /cores/dce/dtsd
- mkdir /cores/dce/dceunixd

最初に、これらのディレクトリーのいずれかに、保存の必要があるデータがあるかどうかをチェックし、必要がある場合には、ログ・ファイルやその他のデータを保存します。

たとえば、

```
mv /opt/dcelocal/var/security/adm/secd/* /cores/dce/secd
```

を、保存したいデータを持つディレクトリーのすべてに対し発行します。

元のディレクトリーは削除します。

- rmdir /opt/dcelocal/var/security/adm/secd
- rmdir /opt/dcelocal/var/security/adm/dceunixd
- rmdir /opt/dcelocal/var/dced
- rmdir /opt/dcelocal/var/directory/cds/adm/gdad
- rmdir /opt/dcelocal/var/directory/cds/adm/cdsd
- rmdir /opt/dcelocal/var/adm/directory/cds/cdsadv
- rmdir /opt/dcelocal/var/adm/directory/cds/cdsclerk
- rmdir /opt/dcelocal/var/time/adm/dtsd

コア・ロケーションを新規の個別のファイルシステムに symlink します。

- ln -s /cores/dce/secd /opt/dcelocal/var/security/adm/secd/
- ln -s /cores/dce/dceunixd /opt/dcelocal/var/security/adm/dceunixd/
- ln -s /cores/dce/dced /opt/dcelocal/var/dced/
- ln -s /cores/dce/gdad /opt/dcelocal/var/directory/cds/adm/gdad
- ln -s /cores/dce/cdsd /opt/dcelocal/var/directory/cds/adm/cdsd
- ln -s /cores/dce/cdsadv /opt/dcelocal/var/adm/directory/cds/cdsadv
- ln -s /cores/dce/cdsclerk /opt/dcelocal/var/adm/directory/cds/cdsclerk
- ln -s /cores/dce/dtsd /opt/dcelocal/var/time/adm/dtsd

セキュリティー・サーバーの再始動

セキュリティー・サーバー (マスターまたはスレーブ) を再始動するためには、次のコマンドを入力します。

マスター:

```
start.dce sec_svr
```

レプリカ:

```
start.dce sec_rep
```

便宜上、マシンがリブートするたびに自動的に始動するように、サーバーを設定します。 (この設定を行うには、 **config.dce** に対して **-autostart yes** オプションを使用します)。

セキュリティー・サーバーの障害後のシステムのバックアップと回復

注: 次のバックアップおよび復元の手順は、既存のサーバー、つまり、 LDAP Migration Server または LDAP Slave Replica Server へ移行していないサーバーだけに適用されるものです。セル内にある LDAP へ移行したすべてのセキュリティー・サーバーで使用するバックアップおよび復元の手順については、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

セキュリティー・サーバーに重大な障害が発生した場合には、次の手順を使用する必要があります。

レジストリーのバックアップ

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

1. マスター・レプリカを保守状態に設定するために、 registry disable コマンドを入力します。次のコマンドは、セル giverny.com の中のマスター・レジストリーを保守状態に設定します。

   ```
   dcecp> registry disable /.../giverny.com/subsys/dce/sec/master dcecp>
   ```

   マスター・レプリカを保守状態に設定すると、マスターは、そのデータベースをディスクに保管し、すべての更新を拒絶します。
2. レジストリーをバックアップするには、 496ページの2a のステップか 496ページの2b のステップを選択します。

   注:

   バックアップにどのようなコマンドを使用すればよいかは、個人の好みの問題です。しかし、データベースとマスター・キー・ファイルの両方を同じテープに書き込む場合には、そのテープは、アクセスが制限された、ロックし

た場所に格納してください。あるいは、データベースとキー・ファイルを別々のテープに書き込み、それぞれのテープを別の場所に格納することもできます。

a. ボリューム全体、または **/opt/dcelocal/var/security/rgy_data** ツリー (レジストリー) と、ファイル **/opt/dcelocal/var/security/.mkey** をバックアップすることによって、マスター・レジストリーをバックアップします。この **/opt/dcelocal/var/security/.mkey** ファイルには、レジストリーの中のすべてのキーの暗号化に使用するマスター・キーが含まれています。ファイル **/opt/dcelocal/var/security/.mkey** にはマスター・キーが含まれているため、ファイル **/opt/dcelocal/var/security/.mkey** もリストアしないかぎり、レジストリー・データベースのバックアップをリストアしても無駄であることに、注意してください。

   以下をバックアップするために tar を使用します。

   ```
   /opt/dcelocal
   /var/dce  (AIX の場合) および /opt/dcelocal/var  (Solaris の場合)
   /krb5
   /etc/dce  (AIX の場合) および /etc/dcelocal/etc  (Solaris の場合)
   ```

b. 別の方法として、以下の dceback コマンドを使用し、重要なセキュリティー・ファイルをバックアップできます。

   レジストリー・ファイルを保管するには、次のようにします。

   ```
   dceback dumpsecurity -destfile <filename>
   ```

   クライアント・ファイルを保管するには、次のようにします。

   ```
   dceback dumpmisc -destfile <filename>
   ```

3. バックアップが完了したら、次のようにマスター・レプリカの保守状態を解除します。

   ```
   dcecp> registry enable  /.../giverny.com/subsys/dce/sec/master dcecp>
   ```

   セキュリティー・サーバーは、更新の受け入れを再開します。

   注: 前の例では、レジストリー 1 次サイトの名前を、 **registry enable** コマンドと **registry disable** コマンドに指定しました。レジストリー・サイト名を指定しない場合には、コマンドは変数 **_s(sec_)** で指定されたサイトを使用します。この変数を設定しなければ、コマンドは、マシンのデフォルトのセルの 1 次レジストリーを使用します。

レジストリーの復元

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

次の手順は、旧セキュリティー・サーバーがもうパワーアップされないことを前提として使用されます。

1. 新しいマシンをネットワークにインストールし、そのマシンに、旧セキュリティー・サーバーと同じ IP アドレスとホスト名を与える。
2. マシンに DCE をインストールするが、構成はしない。

3. ステップ3a かステップ 3b を選択して、レジストリーを復元する。
   a. 保存されているディレクトリーを Untar する。

      /opt/dcelocal
      /var/dce **(AIX の場合)** および /opt/dcelocal/var **(Solaris の場合)**
      /krb5
      /etc/dce **(AIX の場合)** および /etc/dcelocal/etc **(Solaris の場合)**

   b. 別の方法として、次の dceback コマンドを使用し、 dceback コマンドでバックアップしたデータを復元する。

      ```
      dceback restoresecurity -sourcefile <filename>
      dceback restoremisc -sourcefile <filename>
      ```

4. システムを再始動する。

マスター・サーバーのロックスミス・モードでの再始動

secd -locksmith オプションは、**secd** をロックスミス・モードで始動します。このオプションは、マスター・レプリカに対してのみ使用できます。ロックスミス・モードでは、 **secd** に指定したプリンシパル名がロックスミス・プリンシパルになります。ロックスミス・プリンシパルになると、レジストリーへの最高アクセス特権によるログインを妨げる、故意または偶然の変更を修正することができます。

セキュリティー・サーバーをロックスミス・モードにすると、**secd** は、自動的にロックスミス・アカウントを作成するか、あるいはロックスミス・アカウントが存在する場合は、そのアカウントの新規パスワードを与えるよう要求します。セキュリティー・サーバーが実行されると、ユーザーは新しく変更したパスワード (変更した場合) を使用して、ロックスミス・アカウントにログインした上で、レジストリーにアクセスし、通常の証明書によるレジストリーへのアクセスを妨げている可能性のある、アカウントまたはポリシーの情報を変更することができます。

ロックスミス・モードでは、有効なアカウントを持つすべてのプリンシパルは、通常のアクセス検査のもとに、ログインし、レジストリーを操作することができます。しかし、ロックスミス・プリンシパルはレジストリーへの特殊なアクセスを認可されます。すなわち、認証されたロックスミス・プリンシパルについては、アクセス検査は行われません。これは、ロックスミス・プリンシパルになると、最大限のアクセス権によりレジストリーを操作することができることを意味します。

ロックスミス・アカウントへの自動変更

セキュリティー・サーバーをロックスミス・モードで始動するときにロックスミス・アカウントが存在する場合、セキュリティー・サーバーは特定のアカウントとレジストリー・ポリシーの情報を検査して、 498ページの表28 と 498ページの表29 に示す変更を加えます。これらの変更により、アカウントやレジストリー・ポリシーが変更されていても、ロックスミス・アカウントにログインすることが保証されます。たとえば、侵入者がアカウント存続期間のレジストリー・ポリシーを 1 分に変更すると、ロックスミス・アカウントを使用できるだけの有効時間がなくなります。このため、アカウント存続期間のレジストリー・ポリシーが、ロックスミス・アカウントを最低 1 時間有効にするために必要な値よりも小さく設定されていることを、セキュリティー・サーバーが知ると、セキュリティー・サーバーは、アカウント存続期間ポリシーを、ロックスミス・アカウントの作成時刻と現在の時刻から 1 時間後の時刻との差に設定します。

表 28. セキュリティー・サーバーによって行われるロックスミス・アカウントの変更

セキュリティー・サーバーが ... を検出すると、	... に変更します。
パスワード有効フラグが **no** に設定されている	パスワード有効フラグを **yes** に
アカウント有効期限が現在の時刻に 1 時間プラスした時刻より手前に設定されている	アカウント有効期限を現在の時刻の 1 時間後に
クライアント・フラグが **no** に設定されている	クライアント・フラグを **yes** に
アカウント有効フラグが **no** に設定されている	アカウント有効フラグを **yes** に
発効日が現在の時刻より後に設定されている	発効日を現在の時刻に
パスワード有効期限が現在の時刻の 1 時間後より手前に設定されている	パスワード有効期限を現在の時刻の 1 時間後に

表 29. セキュリティー・サーバーによって行われるレジストリー・ポリシーの変更

セキュリティー・サーバーが ... を検出すると、	... に変更します。
アカウント存続時間が、ロックスミス・アカウントの作成時刻と現在の時刻に 1 時間プラスした時刻との差より小さく設定されている	アカウント存続期間を、現在の時刻に 1 時間プラスし、さらにロックスミス・アカウントの作成日をマイナスした時間に
パスワード有効期限が、パスワードの最終変更時刻より後だが、現在の時刻に 1 時間プラスしたより手前に設定されている	パスワード有効期限を現在の時刻の 1 時間後に

ロックスミス・モードでのセキュリティー・サーバーの始動

セキュリティー・サーバーをロックスミス・モードで始動するためには、次の形式の **secd** コマンドを使用します。

dcelocal`/bin/secd [-locksm[ith]` *pname* `[-lockpw] [-rem[ote]]]`

ここで、

-locksm[ith]
セキュリティー・サーバーをロックスミス・モードで始動します。

pname ロックスミス・プリンシパルの名前を指定します。このプリンシパルのレジストリー・アカウントが存在しないと、 **secd** がそれを作成します。

-lockpw
新規のロックスミス・パスワードを入力するようプロンプト指示します。このオプションにより、旧パスワードを知らないときに、ロックスミス・アカウントの新規パスワードを指定することが可能となります。

-rem[ote]
ロックスミス・プリンシパルのリモート・ログインを許可します。このオプションを使用しない場合、プリンシパルは、**secd** を始動するローカル・マシンからログインする必要があります。

ロックスミス・モードでのセキュリティー・サーバーの再始動

ロックスミス・モードでセキュリティー・サーバーを再始動するためには、マスター・レプリカを実行しているノードで次の手順を実行します。このノードへの **root** アクセス権が必要です。

1. セキュリティー・サーバーをシャットダウンする。
   a. 管理特権によりログインし、**dcecp** にアクセスしてサーバーをシャットダウンできない場合は、サーバーを実行しているマシンに **root** としてログインし、セキュリティー・サーバーのプロセスを停止します。
   b. 管理特権によりログインできる場合は、セキュリティー・サーバーをシャットダウンするために、 **dcecp registry stop** コマンドを使用します。このコマンドを使用するときは、停止するレプリカの完全修飾名を引き数として与える必要があります。次のサンプル・コマンドは、 **slave_3** という名前のレプリカを停止させます。

      ```
      dcecp> registry stop  /.../giverny.com/subsys/dce/sec/slave_3
      dcecp>
      ```

2. セキュリティー・サーバーをロックスミス・モードで始動する。次に示す例は、**master_admin** という名前のプリンシパルに対して作成された、ロックスミス・アカウントによりセキュリティー・サーバーを始動する例です。 **master_admin** がリモート・ノードからログインできるように、 **-remote** オプションも指定します。指定しない場合は、**master_admin** は、セキュリティー・サーバーを始動したノードからログインする必要があります。

   ```
   dcelocal/bin/secd -locksmith master_admin -remote
   ```

   ロックスミス・アカウントが存在していながら、そのパスワードを忘れてしまった場合は、 **-lockpw** を使用して、**secd** に新規のロックスミス・パスワードのプロンプトを出させ、入力したパスワードで既存のパスワードを置き換えられるようにします。

セキュリティー・サーバーは通常、バックグラウンドで実行されます。セキュリティー・サーバーをロックスミス・モードで始動すると、ユーザーがプロンプトに応答できるように、それはフォアグラウンドで実行されます。

セキュリティー・サーバーをロックスミス・モードで始動したときは、標準の特権を持つアカウントがレジストリーにアクセスできるように、 **dcecp registry modify** コマンドを使用してレジストリーを変更することができます。これらの変更を行った後、以下のことを行う必要があります。

1. ロックスミス・モードで実行されているセキュリティー・サーバーをシャットダウンする。
2. 標準の手順にしたがってセキュリティー・サーバーを再始動する。

マスター・レプリカの回復

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

マスターのデータベースが損傷しているためにマスター・レプリカを回復する場合は、次のいずれかの方式を使用します。

- **dcecp registry designate** コマンドを使用して、スレーブ・レプリカをマスター・レプリカにし、元のマスターのホスト上にスレーブ・レプリカを作成する。この方式については、次のサブセクションで説明します。
- バックアップからマスターをリストアする。この方式については、 457ページの『第36章 ルーチン保守の実行』で説明します。

どの方式を選択するかは、マスター・レプリカのバックアップ・データベースとスレーブ・レプリカのデータベースのどちらがより新しいかによって決まります。

最新のデータベースの判別

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

マスター・レプリカのデータベースのバックアップと、スレーブ・レプリカのデータベースのどちらがより新しいかを判別するためには、レプリカに対して **dcecp registry show-replica** を実行します。このコマンドの出力は、最後の更新のシーケンス番号と更新日時をリストします。レプリカの最後の更新のシーケンス番号および更新日時を、マスターのバックアップのシーケンス番号および日時と比較します。レプリカの方が新しい場合は、次のセクションで説明するように、レプリカを新しいマスターにします。マスターのバックアップの方が新しい場合は、 457ページの『第36章 ルーチン保守の実行』で説明されているように、バックアップからマスターをリストアします。

スレーブのマスターへの変換

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

このサブセクションでは、スレーブをマスターに変換するための **dcecp registry designate** コマンドの使用方法について説明します。 **registry designate -master** コマンドは、データを失う可能性があるため、異なるマスター・レプリカを指定する方法としては、 **-master** オプションを指定しない **registry designate** コマンドを使用されることをお勧めします。 **registry designate -master** コマンドを使用するのは、マスター・レプリカが回復できないほどの損傷を受けており、 **-master** オプションを指定しない **registry designate** コマンドが使用できない場合に限られます。

マスター・レプリカをスレーブ・レプリカに変更するためには、以下の手順に従ってください。

1. 新しくマスターになるスレーブ・レプリカを選択する。
2. デフォルトのホストをマスター・レジストリーに変更するために、次の **registry designate -master** コマンドを実行する。

```
dcecp>  registry designate/.../musee.com/subsys/dce/sec/art -master
dcecp>
```

3. 変更を確認するために **registry show -replica** コマンドを使用する。
4. 標準の UNIX コマンドを使用して、ディレクトリー *dcelocal*/**var/security/rgy_data** と、ファイル *dcelocal*/**var/.mkey** を削除することによって、旧マスター・レプリカのデータベースと **.mkey** ファイルを削除する。
5. レプリカ・リストから旧マスターを削除するために、 **-force** オプションを指定した **registry delete** コマンドを使用する。次の例では、レプリカ・リストから **history** という名前の旧マスターを削除します。

```
dcecp> registry  delete /.../musee.com/subsys/dce/sec/history -force
dcecp>
```

スレーブ・レプリカの回復

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

スレーブ・レプリカはバックアップされないため、破壊されたレプリカをリストアするためには、レプリカを再作成する必要があります。このためには、以下の手順を使用します。

1. 次の手順を使用してセキュリティー・レプリカ・サーバーの構成を解除する。
   a. **unconfig.dce** コマンドを使用して、**完全**または**部分**構成解除を実行する。

      完全 DCE がセキュリティー・レプリカ・サーバー・マシン上で稼働している場合、次のように使用します。

      ```
      unconfig.dce sec_rep
      ```

      部分 DCE がセキュリティー・レプリカ・サーバー・マシン上で稼働していない場合、次のように使用します。

      - セキュリティー・レプリカ・サーバー・マシンで、次を実行する。

        ```
        unconfig.dce -config_type local sec_rep
        ```

      - DCE セル内の他のマシンで、次を実行する。

        ```
        unconfig.dce -config_type admin -dce_host <sec_rep_machine-dce_hostname> ¥
        -host_id <sec_rep_machine-host_id> sec_rep
        ```

      これらのコマンドにより、セキュリティー・レプリカ・サーバーが完全に終結処理されます。

   b. 前述のステップが機能しない場合、次のコマンドを使用する。
      1) 標準の UNIX コマンドを使用して、レプリカのデータベース・ファイルとマスター・キー・ファイルを手作業で削除する。このためには、次のロケーションのすべてのファイルを削除します。
         - **/opt/dcelocal/var/security/rgy_data**
         - **/opt/dcelocal/var/security/.mkey**
      2) マスターにバインドするために **set _s(sec)** コマンドを使用し、次に、マスターのレプリカ・リストからレプリカを削除するために **dcecp registry delete -force** コマンドを使用する。次の 2 つのコマンドは、マスターにバインドして、次にレプリカを削除する方法を示しています。

```
dcecp> set _s(sec) /.../musee.com/subsys/dce/sec/master
dcecp> registry delete /.../musee.com/subsys/dce/sec/art -force
dcecp>
```

3) 標準の UNIX コマンドを使用して、マスターを実行しているマシンからレプリカを実行するマシンに、ファイル **/opt/dcelocal/etc/security/pe_site** をコピーする。

4) レプリカ・マシンの **/etc/dce_config** (またはプロバイダーのこれと同等のもの) を使用して、以下のことを行う。

a) DCE デーモンを停止する (stop.dce)。

b) DCE デーモンを開始する (start.dce)。

2. セキュリティー・サーバー・レプリカを構成する。この構成により、レプリカ・データベースを作成し、 **secd** を始動します (config.dce)。
3. 前のステップでレプリカを構成するときに、そのレプリカに名前を割り当てる。このレプリカに以前と同じ名前を与えなかった場合は、次の手順を実行して、CDS から前の名前を削除する必要があります。
   a. レプリカのサーバー・エントリーの名前を **/.:/subsys/dce/sec** から削除する。
   b. レプリカの名前を CDS グループ **/.:/sec** から削除する。

マスターのスレーブへの変換

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

マスター・レプリカをスレーブ・レプリカに変更するためには、以下の手順を使用します。この手順を使用するのは、ユーザーのネットワークまたはインターネット上で複数のマスターが実行されている場合に限られます (この状況は非常にまれです)。

1. スレーブになるマスター・レプリカを選択する。
2. 選択したマスターをスレーブに変更するために、次の **registry designate -slave** コマンドを実行する。

```
dcecp> registry designate /.../dublin.com/subsys/dce/sec/lit -slave
dcecp>
```

3. 変更を確認するために **registry show -replica** コマンドを使用する。

スレーブ・レプリカの強制的削除

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

このセクションで記述されている手順は、スレーブ・レプリカを強制的に削除する方法を説明しています。このドラスティックな方法は、 471ページの『第37章 ネットワーク再構成の処理』で説明されている通常の削除方法が失敗したときにだけ使用してください。

スレーブ・レプリカを強制的に削除するためには、 **dcecp registry delete -force** コマンドを使用します。このコマンドは、スレーブ・レプリカをマスターのレプリカ・リストから削除します。その後、マスターは他のレプリカに削除要求を伝搬します。このオペレーションは削除されたレプリカには伝わらないので、 **-force** オプションは、レプリカが使用不能で、再始動できないときにのみ使用します。強制的に削除されたレプリカがオペレーションを継続している場合には、 **registry destroy** コマンドを使用してサーバーを停止し、そのデータベースを削除します。また、(**dcecp registry stop** コマンドを用いて) 単に **secd** を停止して、そのデータベースを削除または名前変更することもできます。

レジストリー・レプリカを強制的に削除するためには、削除するレジストリーの名前を引き数として与え、 **-force** オプションを指定した **registry delete** コマンドを実行します。次の例では、**/.../giverny.com/subsys/dce/sec/lit_server_2** のレプリカを削除します。

```
dcecp> registry delete /.../giverny.com/subsys/dce/sec/lit_server_2
-force
dcecp>
```

デフォルトのレプリカがマスターでない場合、 **dcecp** は自動的にマスターにバインドします。

強制的に削除されたレプリカがオペレーションを継続している場合には、 **registry destroy** コマンドを使用してサーバーを停止し、そのデータベースを削除します。**registry destroy** コマンドを使用するときには、停止したいレプリカの名前を入力する必要があります。次に示す例は、**registry destroy** コマンドを使用して、**/.../giverny.com/subsys/dce/sec/lit_server_2** にあるレプリカを削除する例です。

```
dcecp> registry destroy /.../giverny.com/subsys/dce/sec/lit_server_2
dcecp>
```

代わりに、 (**dcecp registry stop** コマンドを用いて) 単に **secd** を停止して、そのデータベースを削除または名前変更することによりレプリカを削除することもできます。

重複マスターの復元

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

このセクションでは、非常にまれな問題からの回復方法について説明します。絶対必要な場合を除き、問題の解決のためにここで説明する方法を使用しないでください。

ときには、マスターにしたいレプリカのマスター・シーケンス番号が、システム内の別のマスター・シーケンス番号より小さい (または等しい) 場合があります。マスター・シーケンス番号がシステム内の別のものより小さいことをマスターが検出すると、マスターはそれ自体に重複マスターであるというマークを付けて、プロセスを終了します。そのマスター・レプリカを始動するごとに、それは重複マスターと見なされていることを通知して、再びプロセスを終了します。

この重複マスターを強制的にマスターにして、プロセスが終了しないようにするためには、以下に示す形式の、**-master_seqno** オプションを指定した **secd** プロセスを再始動します。

```
secd -master_seqno new_master_seqno
```

ここで、*new_master_seqno* は、レプリカに割り当てる新しいマスター・シーケンス番号です。この番号は、システム内の最大のマスター・シーケンス番号より 1 つ大きくします。

最大のマスター・シーケンス番号を検出するために、 **dcecp registry dump** コマンドを使用します。

現行のマスター・レプリカが失敗したときの新しいマスター・レプリカの指定

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

マスター・セキュリティー・レプリカが失敗し、かなりの間使用できなくなる場合は、次のステップに従って、既存のスレーブ・レプリカを新規のマスター・レプリカとして指定します。

注: これらのステップが終了すると、以前のマスターはセルから構成が解除されます。

1. 新しいマスター・サイトを選択する。スレーブ・レプリカは、ユーザーが選択したホストにすでに存在している必要があります。
2. 新しいマスター・レプリカになるホストにログインします。
3. **/opt/dcelocal/etc/security/pe_site** ファイルを編集して、古いマスター・レプリカの項目をすべて削除します。それから、**TRY_PE_SITE** 環境変数を **1** にセットします。
4. セルの管理者として DCE にログインします。
5. **登録指示**コマンドを出して、新しいマスター・レプリカを設定します。このコマンドを入力する時には、新規マスターにするレプリカの名前を引き数として指定します。この操作では、以前のマスターにはコンタクトしません。このコマンドは、終了するのに数分かかります。

   ```
   dcecp> registry designate /.../henry.com/subsys/dce/sec/cheyenne
   -master
   dcecp>
   ```

6. **registry show** コマンドを出して、新しいマスター・レプリカがマスターになったことを確認する。

   ```
   dcecp> registry show /.../henry.com/subsys/dce/sec/cheyenne
   dcecp>
   ```

7. **registry delete** コマンドを使用して、レプリカ・リストから古いマスター・レプリカを削除する。

   ```
   dcecp> registry delete /.../henry.com/subsys/dce/sec/bourbon -force
   dcecp>
   ```

8. **rpcgroup remove** コマンドを出して、古いマスター・レプリカをセキュリティー rpc groups **/.:/sec** および **/.:/sec-v1** から削除する。

```
dcecp> rpcgroup remove /.:/sec -member /.../henry.com/subsys/dce/sec/bourbon
dcecp>
dcecp> rpcgroup remove /.:/sec-v1 -member /.../henry.com/subsys/dce/sec/bourbon
dcecp>
```

9. **object delete** コマンドを出して、古いマスター・レプリカのための **cds** オブジェクトを cds ネームスペースから削除する。

```
dcecp> object delete /.:/subsys/dce/sec/bourbon
dcecp>
```

10. **acl modify** コマンドを出して、古いマスター・レプリカのホストをレプリカ・リスト ACL から削除する。 (実行している DCE のバージョンによっては、ADL 上に存在しない場合があります。)

```
dcecp> acl modify /.:/sec/replist
-remove {user hosts/bourbon/self -i-m-I}
dcecp>
```

11. **group remove** コマンドを出して、古いマスター・レプリカのホストをセキュリティー・サーバー・グループから削除する。

```
dcecp> group remove /.:/subsys/dce/sec_servers -member /.:/hosts/bourbon/self
dcecp>
```

12. **unconfig.dce** コマンドを入力して、古いマスター・レプリカの管理部分の構成を解除する。

```
$ unconfig.dce -config_type admin -dce_hostname bourbon sec_rep
dcecp>
```

    注: 上のコマンドの **sec_rep** を **all** に置き換えると、古いマスター・レプリカのすべての DCE コンポーネントの構成を解除することができます。

ユーザー間プロトコルで登録済みのサーバーのアクセス

サーバーが停止され、再始動された場合、停止した旧サーバーに対する有効期限内のチケットを持つクライアントは、新規に再始動したサーバーでは認証済み RPC を実行できません。クライアントは、現行のチケットが期限切れになるのを待つか、あるいは、現行の証明書を破棄 (kdestroy) し、新規の dce_login を実行することにより、新しいチケットを取得する必要があります。

レジストリー・オーファンの採用

dcecp はオブジェクト名を表示し、ユーザーは名前でレジストリー・オブジェクトを識別しますが、 DCE セキュリティー・サービスは、UUID を使用して内部でオブジェクトを識別します。レジストリー・オブジェクトを作成するときに、DCE セキュリティー・サービスは、オブジェクトを識別するために使用する UUID とオブジェクト名との間の関連を自動的に設定します。レジストリー・オブジェクトを削除するときには、レジストリー・オブジェクトとそのオブジェクトを識別する UUID の間の関連を削除します。

オーファンは、プリンシパルまたはグループが削除されたために、プリンシパルまたはグループに関連付けられていない、UUID が所有するオブジェクトです。たとえば、レジストリーからプリンシパルを削除すると、プリンシパルを外部的に識別

するために使用した名前とプリンシパルを内部的に識別する UUID との間の関連付けも削除されます。削除されたプリンシパルが所有していたオブジェクト (ファイル、プログラム) は、プリンシパルともはや関連付けられていない 、UUID により内部的に所有されることになります。他のプリンシパル、グループ、またはオーガニゼーションがそのオブジェクトへのアクセス権を持っていない場合は、そのオブジェクトにアクセスすることはまったくできず、そのオブジェクトはオーファン になります。

この問題を解決するために、**dcecp principal create** を使用することができます。 **group create** および **org create -uuid** コマンドを、**-uuid** オプションを指定して使用し、オーファンのオブジェクトを所有する UUID と同じ UUID を持つ、プリンシパル、グループ、またはオーガニゼーションを作成し、そしてそのオーファンのオブジェクトを採用します。

注: 新しいレジストリー・オブジェクトを作成するときは、そのオブジェクトと関連した UUID を指定する方法はありません。したがって、オーファンを採用するために、単に同じ名前の新しいレジストリーを追加することはできません。

-uuid オプションを指定すると、プリンシパル、グループ、またはオーガニゼーションが作成され、それに関連付ける必要がある UUID を指定することになり、自動的割り当ては行われません。作成の方法を除けば、それらのコマンドで作成されるプリンシパル、グループ、またはオーガニゼーションは、他のプリンシパル、グループ、またはオーガニゼーションと異なりません。次の例では、オーファンになったレジストリー・オブジェクトを採用するために、このオプションを使用してプリンシパル、グループ、またはオーガニゼーションを作成する方法を示しています。

オーファンになったオブジェクトを所有している UUID と関連したプリンシパルを作成するためには、次のコマンドを使用します。

```
principal create name -uuid uuid [-fullname fullname] ¥
     [-quota object_creation_quota] [-uid UNIX_number]
```

オーファンになったオブジェクトを所有している UUID と関連したグループを作成するためには、次のコマンドを使用します。

```
group create name -uuid uuid [-fullname string] ¥
     [-inprojlist [yes| no]] [-gid UNIX_number]
```

オーファンになったオブジェクトを所有している UUID と関連したオーガニゼーションを作成するためには、次のコマンドを使用します。

```
organization create name -uuid uuid ¥
[-fullname string] [-orgid UNIX_number]
```

ここで、

name プリンシパル、グループ、またはオーガニゼーションの 1 次名。

uuid プリンシパル、グループ、またはオーガニゼーションに割り当てられる UUID 番号。この UUID は、オーファンになったオブジェクトを所有している UUID (つまり、削除されたレジストリー・オブジェクトに関連した UUID) である必要があります。 UUID は、8 桁の 16 進数、ハイフン、4 桁の 16 進数、ハイフン、 4 桁の 16 進数、ハイフン、4 桁の 16 進数、ハイフン、および 12 桁の 16 進数という、RPC 印刷ストリング形式で指定されます。その形式は次のとおりです。

nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn

string プリンシパル、グループ、またはオーガニゼーションのフルネーム。

UNIX_number
セル・プリンシパルの場合、名前と関連付ける、 UNIX 番号のみ。このオプションを入力しないと、次の順番の UNIX 番号が与えられます。セル以外のすべてのプリンシパルの場合、 UNIX 番号はプリンシパルの UUID に組み込まれた情報から抽出され、ここでは指定できません。

object_creation_quota
プリンシパルの場合、プリンシパルのオブジェクト作成割り当て量のみ。このオプションを入力しなければ、デフォルトは **unlimited** です。

-inprojlist
グループの場合、**yes** のみ が、プロジェクト・リスト包含をオフにし、グループはプロジェクト・リストに含められません。 **no** を入力すると、そのグループはプロジェクト・リストに含まれます。

注: DCE の現行の実装では、UNIX 番号は UUID に組み込まれています。オーファンになったオブジェクトを採用するために、グループまたはオーガニゼーションの作成を試行して失敗した場合、その原因としては、組み込まれた UNIX 番号が、レジストリー属性としてセルに対して指定された有効な UNIX 番号の範囲内に入らないために、無効になっていることが考えられます。この場合は、UUID に組み込まれた UNIX 番号を含むように、有効な UNIX 番号の範囲をリセットし、再びオブジェクトの採用を行ってみる必要があります。
UNIX 番号の有効範囲の設定については、 447ページの『第35章 ポリシーと属性の保守』 を参照してください。

第41章 AIX/DCE セキュリティーの統合 (AIX のみ)

このリリースでは、AIX 基本オペレーティング・セキュリティー・サービスが、DCE セキュリティー・サービスに統合されました。この統合は、ローカル AIX システムとリモート DCE システムという別々のイメージではなく、単一システム・イメージを一般のエンド・ユーザーに提供するように設計されています。「詳細情報 - セキュリティー統合」で説明されているように、この統合には多少の制限があります。しかし、大部分のユーザーは、AIX コマンド (**login** と **su**) により DCE 証明書を取得し、 AIX **passwd** コマンドにより DCE パスワードを変更し、さらに、標準の AIX **libc.a** ルーチン **getpwnam()**、**getpwuid()**、 **getgrnam()**、および **getgrgid()** をとおして、 DCE レジストリーから AIX タイプ情報をリトリーブすることができます。より一般的なルーチンである、**getpwent() と getgrent()** は、現時点では DCE に知られていません。

詳細情報 - セキュリティー統合

DCE レジストリー、および個々のシステムの **/etc/passwd と /etc/group** ファイル内のユーザーとグループは、できるだけ緊密に同期化することを強くお勧めします。同期化は、初期のセルの構成後、**passwd_export** および **passwd_import** ユーティリティーにより実施できます。セル・レジストリーとローカル・ファイルの間で同期化されていないユーザーは、統合機能の利点を完全には実現できません。一方、この柔軟な統合方式は、放浪のユーザー (DCE レジストリーでは定義されているが、ローカル・システムでは定義されていないユーザー) をサポートします。マシンの構成が許す場合、そのような放浪のユーザーは、システムにログオンし、レジストリーの UNIX 関連情報に基づいて、DCE 証明書とローカル・アクセスを取得することができます。

非統合の DCE セキュリティー・コマンド **dce_login** (DCE へのロギング用) と **dcecp** プログラム (DCE パスワード変更とレジストリー照会用) は、以前と変わらずに使用できます。もちろん、ローカル・アクセスはこれらのコマンドを使用するための前提条件です。

ローカル管理者は、AIX/DCE のセキュリティー統合を可能にするように、システムをセットアップする必要があります。この管理者は、このセクションで述べる手順をローカル・ユーザー **root** として実行する必要があります。システムは、DCE クライアント・マシンとして構成済みである必要があります。

DCE アクセスの使用可能化

/etc/security/user ファイルは、特定のユーザー (スタンザを持つ) とその他のユーザー (デフォルトのスタンザを使用する) の両方についての属性を定義します。AIX/DCE セキュリティー統合をサポートするために、2 つの新しいユーザー属性、**-SYSTEM** と **registry** が定義されています。特定のユーザーまたはシステム全体に対して DCE アクセスを許可または否認するために、 **SYSTEM** 属性を使用します。パスワード操作をローカルまたは DCE のいずれかに強制するために、**registry** 属性を使用します。

これらのユーザー属性およびその他のユーザー属性は、ドメイン関連の名前に基づいて適用されることに注目する必要があります。すなわち、**jane**、**/.:/jane**、または **/.../this_cell/jane** としてシステムにログオンする放浪の DCE ユーザーは、**/etc/security/user** のローカル・ユーザー **jane** に対するスタンザの中に存在することがある、すべてのユーザー属性によって影響されます。

SYSTEM 属性

SYSTEM 属性は、ユーザーをシステムに認証するために使用されるメソッドを記述します。この属性に対する有効な値は、次のようなトークンで構成される、ブール式ストリングです。

ファイル
ローカル **/etc/passwd** ファイルのみによる認証。

compat
ローカル・ファイルまたは NIS データベース (あるいはその両方) による認証。

DCE DCE レジストリーによる認証。

SYSTEM 属性の値は、上記のメソッド (トークン) と **AND** および **OR** ブール演算子で構成される、複合式であることがあります。さらに、**SYSTEM** 式は、メソッドからの異なる結果を指定することができます。その構文は `method [result]` です。有効な結果は、 `UNAVAIL` (認証サービスが選択不可であった)、 `NOTFOUND` (データベース内でユーザーが検出されなかった)、 `FAILURE` (その他の未指定の理由で、認証が失敗した)、または `SUCCESS` (メソッドにおいて暗黙的で結果を伴わない) などです。

デフォルトのスタンザは `SYSTEM = compat` で作成されます。したがって、ローカル管理者は、このスタンザまたは特定のユーザー・スタンザを、別の認証方式を必要とするユーザーのために変更する必要があります。ローカル・ユーザー root は、常に `SYSTEM = compat` (および `registry = files`) を持つ必要があります。したがって、スーパーユーザーは非ローカル認証メカニズムによって左右されません。

SYSTEM ユーザー属性とその有効な値の詳細な解説については、 AIX の資料を参照してください。

registry 属性

registry 属性は、ユーザーのパスワードを管理するデータベースを定義します。この属性は、パスワード照会とその変更が行われる場所を決定します。この属性に対する有効な値は次のとおりです。

ファイル
ユーザーのパスワードはローカル・ファイルで管理されます。

NIS ユーザーのパスワードは NIS データベースで管理されます。

DCE ユーザーのパスワードは DCE レジストリーで管理されます。

registry にはデフォルト値はなく、後で説明するように、管理者はデフォルトを未定義のままにしておくこともできます。ローカル・ユーザー **root** は、常に、

registry = files (および SYSTEM = compat) として定義する必要があります。したがって、スーパーユーザーのパスワード操作は、非ローカル・メカニズムに左右されません。

registry ユーザー属性とその有効な値の詳細な解説については、 AIX の資料を参照してください。

ローカル・リソースの保護

ローカル・システム管理者はすべてのローカル・リソースを保護する必要があります。すでに述べたように、DCE レジストリーとローカル **/etc/passwd** ファイルおよびローカル **/etc/group** ファイルは、できるだけ緊密に同期化することを強くお勧めします。しかし場合によっては、ローカル・ユーザーをレジストリーと同期化しない方が望ましいこともあります (ローカル・ユーザー **root** が良い例です)。その場合には、ローカル管理者は、すべての非同期化名または非同期化 ID (あるいはその両方) が保護されているように注意しなければなりません。

ローカル管理者は、分かっているすべての名前 (ユーザー名またはグループ名) あるいは ID (UID または GID) が、 DCE レジストリーと同期化しないように保護し、それらの名前がローカル・システム上にデータを持つことができるように、処置する必要があります。たとえば、ユーザー **joe** がローカル・マシン上に存在し、別のユーザー **joe** が DCE レジストリーに存在する場合、ローカル管理者は、DCE **joe** がそのマシンへアクセスできないことを確認する必要があります。

次のセクションでは、ローカル・リソースを保護するためのさまざまな方法について説明します。

/etc/security/user ファイルによるローカル・リソースの保護

システム内の同じ名前のユーザーと同期化すべきでないことが分かっているユーザーは、すべて、**/etc/security/user** ファイルの **SYSTEM** 属性と **registry** 属性により、ローカル (または NIS) として定義される必要があります。

すべての放浪の DCE ユーザーがローカル・システムにアクセスできないようにするために、デフォルト・スタンザは、**SYSTEM** 属性から DCE を除外する必要があります。

```
default:          SYSTEM = compat
                  registry = files
```

DCE レジストリーと同期化しているローカル・ユーザーは、その個々のスタンザに認証メソッドとして DCE を指定することにより、ログイン時に DCE 証明書を獲得することが依然として可能です。

passwd_override ファイル と group_override ファイルによるローカル・リソースの保護

標準の DCE **passwd_override** 機能は、統合認証では有効です。すなわち、ローカル管理者は、**/opt/dcelocal/etc/passwd_override** ファイルに適切なエントリーを入れることにより、 DCE 認証がユーザ名、ID、または GID に基づいて継続しないようにすることができます。たとえば、次の **passwd_override** エントリーにより、ユーザー **joe** による DCE 認証はできなくなります。

```
joe:OMIT:::::
```

同様の UID または GID をキーとする **passwd_override** エントリーは、指定された ID を持つユーザーによる DCE 認証をできなくします。

基本 DCE の場合、 **/opt/dcelocal/etc/group_override** ファイルは、**passwd_export** オペレーションの間に作成されたグループ・エントリーの変更のために使用されます。 AIX/DCE 統合セキュリティー機能を持つシステムにおいては、 **group_override** ファイルは、DCE ユーザーのローカル・アクセスを妨げるためにも使用できます。そのためには、**group_override** エントリーにグループ名と OMIT を指定します。それにより、そのグループのメンバーはローカル・アクセスを否認されます。たとえば、

```
temps:OMIT::
```

は、**temps** グループに属するすべての DCE ユーザーがローカル・システムにロギングすることを妨げます。 **temps** が、ユーザーのグループ・セットの中で 1 次グループであるか、任意のグループであるかは関係ありません。 **group_override** ファイルによるアクセス否認は、統合 BOS 機能 (**dce_login** のような DCE 独自の機能ではない) の場合にのみ有効であり、否認はグループ名に基づいてのみ行われます。 GID に基づいてアクセスを否認するためには、 **passwd_override** ファイル・メカニズムを使用します。

このリリースでは、**dced** デーモンはオーバーライド機能を実装しています。オーバーライド機能はローカル・リソースの保護に不可欠であるため、 **dced** デーモンが選択不可であると、DCE 認証は失敗するように設計されています。ローカル管理者は、必要な場合は、他の認証方法によってもローカル・アクセスが可能であることを確認する必要があります。 **passwd_override** ファイルと **group_override** ファイルの全体の説明については、 DCE for AIX Administration Command Reference を参照してください。

注: dced は、スリムなクライアント構成上では利用不能なので、**passwd_override** 機能と **group_override** 機能もスリムなクライアント構成上では利用不能です。

注: パフォーマンス上の理由で、ローカル・リソースを DCE オーバーライド・ファイルで保護するよりも、 **/etc/security/user** ファイル (可能な場合) で保護する方が賢明です。

ユーザー単位での DCE アクセスの構成

以下のセクションでは、ユーザー単位でのアクセスの構成について、特に詳細に説明します。

ローカル Root ユーザーの構成と保護

ローカル・スーパーユーザーの認証は、非ローカルなメカニズムによって異なるべきではありません。したがって、 **/etc/security/user** は、常に次のようなエントリーを持つ root スタンザを含む必要があります。

```
root:         SYSTEM = compat
              registry = files
```

このようにして、すべての認証操作とパスワード操作はローカル・ファイルに向けられます。これにより、DCE ユーザー **root** がこのシステム上の DCE にロギングすることも妨げられます (ただし、偶然、DCE ユーザー **root** のパスワードが root パスワードと一致したときは除きます)。

また、UID **0** を持つその他の DCE ユーザーがシステムにロギングすることを防止するために、 **passwd_override** 機能を使用することもできます。これは、放浪の DCE ユーザーの認証をシステム上で可能にし、セル管理者が DCE ユーザー **root** の別名を作成した場合にのみ必要です。別名が存在するかどうかは、 **dcecp** で DCE レジストリー内のアカウントを表示することによって判別できます。

ローカル専用ユーザーの構成と保護

ローカル専用ユーザーとは、ローカル・システム上で定義されているが、 DCE レジストリー内のどのプリンシパルとも同期化されていないユーザーのことです。したがって、その名前または ID がローカル・ユーザーと一致してしまう DCE ユーザーもありえます。そのようなユーザーは、 **SYSTEM** 属性と **registry** 属性を定義する **/etc/security/user** に、次のような特定のスタンザを持つ必要があります。

```
joe:
        SYSTEM = compat
        registry = files
```

注: 該当する場合は、**SYSTEM** を **files** に設定し、 **registry** を **NIS** に設定することができます。その目的は、その属性から DCE を明白に除外することにあります。

したがって、ローカル・ユーザー **joe** は、認証操作またはパスワード操作について、 DCE に依存しません。また、これによって、どの DCE ユーザー **joe** もシステム上の DCE にロギングできなくなります。

また、UID または GID をキーとする **passwd_override** エントリーを作成するか、グループ名をキーとする **group_override** エントリーを作成することにより、ローカル専用ユーザーのシステム・リソースを保護することもできます。

同期化ユーザーの構成

うまく統合が行われたセルにおいては、大部分のユーザーは、 DCE レジストリー内で定義と管理が行われ、ローカルに存在するユーザーは、 DCE ユーザーと同期化する必要があります。パスワードを除く、すべてのローカル・ユーザーの情報を DCE レジストリーと同期化するために、 **passwd_export** ユーティリティーを使用することができます。次に、パスワードを同期化するために、 AIX **passwd** コマンドまたは **pwdadm** 管理コマンドを使用する必要があります。

同期化したユーザーは、AIX BOS コマンドを使用することができ、それらのコマンドを DCE レジストリーに指示することができます。このために、 **/etc/security/user** 内のユーザー・スタンザ (特定のスタンザかデフォルトかに関係なく) は、試行する最初の認証として DCE を定義する **SYSTEM** 属性を持つ必要があります。次に例を示します。

```
      SYSTEM = "DCE OR (DCE [UNAVAIL]  AND compat)"
OR
      SYSTEM = "DCE OR (DCE FAILURE]  AND compat)"
OR
      SYSTEM = "DCE AND compat"
```

注: 同期化したユーザーの場合は、レジストリー属性は設定しません。

上記の例では、DCE が試行される最初の認証メソッドです。認証がパスすれば、ユーザーは次のアクセス権を与えられます。 UNIX タイプのすべての情報 (UID、GID、ホーム・ディレクトリー、ログイン・シェル)、および DCE 証明書が DCE をとおして取得されます。最初の例で、選択不可のために DCE が失敗すると、ローカル認証が試行されます。 2 番目の例は、 DCE 認証が何らかの理由で失敗した場合、ローカル認証を試行します。 3 番目の例では、ユーザーがシステムへのアクセスを許可されるためには、その前に DCE 認証とローカル認証の両方が成功しなければなりません。

ユーザーの **AUTHSTATE** 環境変数は、続いて行われる最初の認証メソッドに設定され、後続の操作は **AUTHSTATE** で定義されたレジストリーに指示されます (このセクションの「アクセス方式の識別」を参照してください)。

同期化したユーザーに対しては **registry** 属性は設定しないでください。パスワード操作を指示すべき適切なレジストリーについて、 **AUTHSTATE** 環境変数が診断されます。言いかえれば、**AUTHSTATE** が **DCE** であると、**registry** は **DCE** であり、 **AUTHSTATE** が **files** または **compat** であると、**registry** は **compat** です。 registry 属性が、ユーザーの **AUTHSTATE** と競合するレジストリーに明示的に設定されると、パスワード操作は失敗することがあります。

放浪の DCE ユーザーのサポート

放浪の DCE ユーザーとは、DCE レジストリーでは定義されているが、ローカル・マシン上では定義されていないユーザーのことです。次のようにデフォルトの **SYSTEM** 属性を設定することにより、放浪のユーザーがどのマシンからでも DCE にログオンできるようにすることができます。

```
default:
        SYSTEM = "DCE OR compat"
```

どの DCE ユーザーも、DCE の名前とパスワードを指定してこのように構成されたマシンから DCE にログオンできます。もちろん、競合する名前または ID を持つローカル・ユーザーが存在し、 **passwd_override** ファイルまたは **group_override** ファイル内の、その独自のスタンザ・エントリーで保護されている場合には、放浪のユーザーはシステム・アクセスを否認されます。

アクセス方式の識別

認証後、**AUTHSTATE** 変数がユーザー環境で設定されます。この変数は、システムへアクセスするためにユーザーにより実装された認証メカニズムを定義します。また、この変数は **files**、**NIS**、または **DCE** などの値を持つことができます。**AUTHSTATE** 変数は、ユーザーの次の操作のためにアクセスされる、最初のデータベースを判別します。しかし、ユーザーの **registry** 属性が定義されている場合、パスワード操作は、ユーザーの **AUTHSTATE** 属性には関係なく、 **registry** によって定義されたデータベースに指示されます。ローカル管理者は、ユーザーの **registry**

属性が、 **SYSTEM** 属性によって間接的に決定される、ユーザーの **AUTHSTATE** と競合しないことを確認する必要があります。 1 つの方法は、**registry** 属性を、ローカル専用ユーザーを除く、すべてのユーザーに対して未定義のままにしておくことです。ローカル専用ユーザー (**root** など) は、常に `registry = files` として定義してください。

パスワードの変更

パスワード操作は、**registry** ユーザー属性で定義されたレジストリー、または、レジストリー属性定義が欠如している場合は、 **AUTHSTATE** 環境変数により定義された**レジストリー**に向けて指示されます。パスワード操作はローカル・レジストリーと DCE レジストリーの両方に指示されることはありません。 DCE 専用ユーザーのパスワード変更は、**dcecp** を介して行われますが、同期化されたユーザーのパスワード変更 (DCE とローカルの両方) は、 2 つのステップからなる手順で AIX **passwd** コマンドにより行うことができます。

```
$ AUTHSTATE=DCE passwd
$ AUTHSTATE=compat passwd
```

ユーザーは、変更前に **AUTHSTATE** 値をエコーし、変更完了後に、元の値に設定し直す必要があります。

注: 同期化しているユーザーについては、パスワードも同期化を保持する必要があります。パスワードの同期化が保持されていないと、 DCE 認証またはローカル認証のいずれかが失敗します。また、ユーザーが複数のマシン上にローカルに存在する場合、ローカル・パスワードはすべてのマシンで同期化している必要があります。

トラブルシューティング

以下では、発生の可能性があり、管理者の注意を必要とする、いくつかの状況について説明します。 BOS コマンドに関係する問題はいずれも、システム・ユーザーにも悪い影響を与える可能性があります。

dceunixd が次のメッセージを出して、始動しません。 (dceunixd) `Ensure that valid machine credentials exist and that the` **`KRB5CCNAME`** `environment variable is not set.`

原因: 既存の証明書 (**KRB5CCNAME** 環境変数によって判別される) が存在しない場合に発生することがあります。

アクション: KRB5CCNAME が環境に設定されている場合、設定を解除します。また、マシン証明書 **(/opt/dcelocal/var/security/cred/dcecred_ffffffff)** が存在することも確認します。新しいマシン証明書を作成するために、 **dced** を始動 (または再始動) する必要があることもあります。

dceunixd が、次のメッセージを出して、始動しません。 (dceunixd) `Detected another instance of dceunixd, pid XXXXX` **(ここで XXXXX はプロセス ID を表す)。**

原因: dceunixd がすでにマシン上で稼働している場合、これが発生することがあります。

アクション: 指摘されたプロセス ID を持つ **dceunixd** プロセスが、稼働中であることを確認します。稼働中の場合は、2 番目の **dceunixd** は始動できません。そのようなプロセスが存在しない場合は、次のファイルを削除します。

```
/opt/dcelocal/var/security/adm/dceunixd.skt
/opt/dcelocal/var/security/adm/dceunixd.pid
```

次に、**dceunixd** を再起動します。

dceunixd プロセスが非常に大きくなります。

原因: これは、マシン上に多数の認証要求が出されたときに発生する可能性があります。それぞれの DCE 認証で認証プロセスが大きくし、また、**dceunixd** が多くのクライアントに代わって認証するので、極めて大きくなる (システム稼働状況により異なる) ことがあります。

アクション: dceunixd が大量のメモリーを消費するためにシステム・パフォーマンスが低下した場合は、デーモンを停止して、再始動します。 **dceunixd** が定期的に停止、再始動される場合は、 **cron** ジョブの実行を検討する必要があります。

AIX コマンドが、次のメッセージを出して、始動しません 。 `Security method "DCE" could not be loaded.`

原因: これは、AIX/DCE セキュリティー統合のためにマシンが適切に構成されていない場合に発生する可能性があります。

アクション: ローカル **dceunixd** デーモンが稼働していて、モジュール **/usr/lib/security/DCE** がマシンにインストールされていることを確認します。次の所有権と許可を持っている必要があります。

```
-rwxr-xr-x  1 root    system
```

その他の構成情報については、「手順 － セキュリティー統合」を参照してください。

ls -l コマンドが、UNIX ID を間違ったユーザー名またはグループ名にマップするか、または ID を名前へマップすることに失敗します。

原因: これは、 **AUTHSTATE** 環境変数が予期しないデータベースをポイントすると発生する可能性があります。

アクション: AUTHSTATE 環境変数をチェックします。すべての **getpw*** および **getgr*** 操作 (**getpwent()** と **getgrent()** を除く) は、最初 **AUTHSTATE** で指示されたデータベースに向けて指示されます。したがって、`AUTHSTATE = DCE` の場合は、照会は DCE レジストリーへ行きます。名前または ID が DCE で検出されない場合は、ローカル/NIS データベースを照会します。代わりに、`AUTHSTATE = compat` の場合は、 DCE より前にローカル・データベースを照会します。したがって、照会操作の動作が、ID から名前のマッピング・コマンド (**ls -l** または **id** など) の出力に影響を与えることがあります。ミス・マッピングまたはあいまいさが生ずるのは、ローカル・ユーザー ID が DCE ユーザ ID と競合する (同期化されていないユーザー) システム上においてのみです。

認証 (たとえば、su、login、telnet、ftp) が失敗しました。

原因: これは、システムとユーザー構成次第で、さまざまな理由で発生する可能性があります。

アクション:

1. ユーザーの **SYSTEM** 属性をチェックします。 **/etc/security/user** のユーザー (またはデフォルト)・スタンザは正しい認証メカニズムを指定していなければなりません。同期化したユーザー (ローカルと DCE の両方で定義されたユーザー) の場合は、 **SYSTEM** 属性は、DCE 認証が失敗してもローカル専用の認証が行われるように定義する必要があります。
2. 放浪のユーザーは別のマシン上で認証を試行する必要があります。ローカル・システム上の明示的保護手段のためにアクセスが否認されることがあります。
3. **dced** が稼働していることをチェックします。 **dced** が利用不能であると、DCE 認証は失敗します。

レジストリー・オブジェクトが検出されませんでした

原因: コマンドは実際には正常に完了したかもしれません。

たとえば、セキュリティー操作がデフォルトで **DCE OR compat** に向けて指示される (**/etc/security/user** のデフォルト・スタンザ内の **SYSTEM** 属性が **DCE OR compat** に等しい) ように構成されているマシンを考えてみてください。そのマシン上で **su** をローカル専用ユーザーに試行すると、ログインの過程で DCE レジストリーが照会されます (その理由は、**SYSTEM** 属性で指定された最初の認証メカニズムであるからです)。正当な「not found (検出されませんでした)」というメッセージが DCE から戻され、 **su** プロセスは、その後に続く次の認証メソッド (**compat**) へ進む前に、そのメッセージを表示します。

アクション: ユーザーの場合、認証が行われたことの妥当性検査のために、 **id** コマンドを実行します。システム管理者の場合、**/etc/security/user** ファイルを開いて、特定のユーザーに使用する認証メカニズムを決定します。

どのシステムでも、セキュリティー操作は最初 DCE に対し指示され、次にローカル・システムに対して指示されるように構成できることを、ユーザーは知っている必要があります。したがって、ローカル専用ユーザーにおける操作 (**su** および **login** など) は、ローカル・ケースに成功する前に、DCE エラー・メッセージを表示することになります。

統合ログイン障害が発生しました。

原因: DCE の応答時間が非常に遅い状況では、統合ログインに障害が発生している可能性があります。統合ログインの処理中に、数多くの異なった照会が DCE レジストリーに対して行われます。そうした照会のどれかが時間内に終わらない場合、ログイン処理はタイムアウトになり、失敗します。

大量のネットワーク通信量、WAN のスピード、 DCE レジストリーから戻されているデータのサイズなどの変数によって、障害が起こる限界値が決まります。変数のこの組み合わせは、これ以外にも DCE のパフォーマンス上の問題を引き起こす可能性があります。

アクション: どのグループにも最大で 1000 以上のユーザーが入らないようにレジストリーを構成したり、どのユーザーも最大で 1000 以上のグループのメンバーにならないようにレジストリーを構成したりすることにより、統合ログインに障害が起こる可能性を減らすことができます。このような構成にすることにより、ログインの試行時に処理されるデータ量が制限されます。

DCE 障害は統合セキュリティー操作に影響を及ぼす場合があるため、必要に応じてシステムがローカル専用のオペレーションにフォールバックしているかどうか確認する必要があります。

手順 - セキュリティー統合

統合セキュリティー操作のためのシステムを構成する場合、次の手順を使用します。

1. モジュール **/usr/lib/security/DCE** が、マシンにインストールされていることを確認する。
2. 次に示す行を含むように、**/etc/security/login.cfg** ファイルを編集する。

```
DCE:
        program = /usr/lib/security/DCE
This defines the authentication method DCE to the system.
```

3. デーモン **dceunixd** がマシンで稼働していることを確認する。このデーモンは、BOS コマンドに代わって、DCE サーバー **secd** および **dced** に連絡します。詳細については、DCE for AIX Administration Command Reference の **dceunixd** 参照ページを参照してください。
4. ユーザーに対し DCE アクセスの許可または否認 (あるいはその両方) を行うために、 **/etc/security/user** スタンザ・ファイルを編集する。このファイルの編集に関する詳細については、「詳細情報 - セキュリティー統合」の「DCE アクセスの使用可能化」を参照してください。
5. ある種のユーザーによる DCE へのアクセスを明示的に妨げるために、**/opt/dcelocal/etc/passwd_override** ファイルと **/opt/dcelocal/etc/group_override** ファイルを作成または編集する。このファイルの編集に関する詳細については、「詳細情報 - セキュリティー統合」の「ローカル・リソースの保護」を参照してください。

制約事項 - セキュリティー統合

このリリースには、統合機能とそれを利用するユーザーに関して、いくつかの制約事項があります。その制約事項は以下のとおりです。

- AIX 基本コマンド (**login**、**su**、または **telnet** など) により取得される DCE 証明書は、 DCE の意味体系を保持します。すなわち、DCE 証明書は、DCE レジストリー・ポリシーが決めたとおりに有効期限が切れ、ローカル・ログイン・セッションの間も継続するということはありません。 **rmxcred** を実行するときは注意する必要があります。すなわち、ログイン時に設定され、多くのウィンドウでアクティブな証明書は削除されることがあり、整合性のある (ウィンドウ間で) DCE 証明書を回復するためには、完全なログインを行う必要があります。
- DFS を使用している場合には、**dceunixd** の始動の前に、マシン上で DFS クライアントを構成し、実行する必要があります。ユーザーが DFS の実行前に DCE

証明書を取得するために、内蔵された **login** または **dce_login** を動かそうとしても、 DFS ファイルへの認証アクセスを得ることができません。

- 8 文字を超える長さのセル相対名を持つ DCE ユーザーは、いずれも統合機能を利用することはできません。 8 文字の制限は UNIX の規制です。
- スラッシュ (/) を含むセル相対名を持つ DCE ユーザーは、統合機能を利用することはできません。したがって、 DCE ユーザー **mary** は統合機能を利用でき、DCE ユーザー **writers/mary** は統合機能を利用できません。
- ローカル・セル・ユーザーのみがサポートされます。外部プリンシパルは統合操作を使用できません。外部プリンシパルは、セル間アクセスについては、**dce_login** と **dcecp** を使用することができます。
- 放浪の DCE ユーザーにログインを許可し、ローカル・ユーザーが十分には保護されていないシステムにおいては、ローカル・ユーザーに対する一部の AIX 生成データが間違って表示されることがあります。このデータには、システム監査証跡の最後のログイン時間と情報が含まれます。同じ名前を持つ DCE ユーザーによりシステムがアクセスされると、ローカル・ユーザーのための情報が影響を受けます。たとえば、ローカル・システムがユーザー **joe** を定義し、しかも、このユーザーが DCE ユーザー **joe** と異なるという、シナリオを想定してみてください。ローカル・システムと DCE レジストリーの UID または GID の間には、競合はないものとします。ローカル管理者は、DCE ユーザー **joe** によるログインの指定変更 (**passwd_override** ファイルをとおして) に失敗し、放浪のユーザーが、 **/etc/security/user** のデフォルト **SYSTEM** 値を `compat OR (compat [FAILURE] AND DCE)` に設定することにより、ログインできるようにしてしまいました。

  DCE ユーザー **joe** は、このシステムにログインすると、アクセス権を獲得し、DCE レジストリーから取得した UID と GID を所有します。したがって、DCE **joe** は、ローカル **joe** のファイルがローカル **joe** の UID と GID に所有されているために、そのファイルにアクセスすることができません。しかし、ローカル・システムは、それでもユーザーを **joe** として認識し、そのユーザーが、ある状況の場合に属するドメイン (DCE またはローカル) を識別することができません。したがって、ログに記録されるデータは、 DCE **joe** またはローカル **joe** のどちらがそのデータを生成したかに関係なく、 **joe** に属されることになります。

  管理者はこの問題をガードするために、可能性のあるすべてのローカル・ユーザーを DCE レジストリーと同期化し、同期化されないユーザーがある場合は、(競合する名前および ID にはローカル・オーバーライドを適用し、**/etc/security/user** スタンザを適切に構成することによって、) すべて適切に保護する必要があります。
- ローカル **root** ユーザーは、DCE ユーザーへの **su** および DCE 証明書の取得はできません。その理由は、ローカル **root** ユーザーは、 DCE ログインの場合に必要な、パスワードをプロンプト指示されないからです。事実、パスワードをプロンプト指示しないコマンド (**.rhosts** ファイルが存在するときの **rcp**、**rsh**、または **rlogin** など) はいずれも、 DCE 証明書の獲得のために使用することはできません。 DCE 証明書を獲得する BOS コマンドは、ユーザー情報とパスワード情報のために DCE レジストリーを使用するという点で DCE を知っています。BOS コマンドは、基礎的な Kerberos 認証プロトコルは使用しません。
- AIX **passwd** コマンドは、その **pwd_val_type** 拡張レジストリー属性がユーザー選択のパスワードを指定する、 DCE ユーザーについてのみ DCE パスワード

を変更します。このコマンドは、その **pwd_val_type** era (拡張レジストリー属性) がシステム生成パスワードを指定している、 DCE ユーザーについてはパスワードを変更できません。

- ローカル DCE レジストリーは DCE 認証をパスするのに十分ではありません。DCE セキュリティー・サーバーが選択不可で、しかも DCE が **SYSTEM** 属性によって必須認証メソッドとして定義されている場合は、ユーザーはシステムにアクセスすることができません。
- パスワード操作は、DCE レジストリーまたはローカル・ファイルのいずれかに指示され、両方に指示されることはありません。したがって、DCE とローカル・システムの両方に存在するユーザー (同期化したユーザー) のパスワード変更手順は、ステップが 2 つになります。 AIX **passwd** コマンドを使用して、最初に 1 つのレジストリーのパスワードを変更し、次に別のレジストリーのパスワードを変更します。たとえば、次のとおりです。

```
$ AUTHSTATE=DCE passwd
$ AUTHSTATE=compat passwd
```

/etc/passwd 情報、または **/etc/group** 情報に依存する一部の AIX BOS コマンドは、DCE を知りません。それに該当するコマンドとしては、 **finger**、 **quot**、 **lsuser** と、いくつかの管理コマンド (**mkuser**、 **rmuser**、 **mkgroup**、 および **rmgroup** など) があります。

例 - セキュリティー統合

- ユーザー **joe** (そのローカル **/etc/passwd** 情報が DCE レジストリーと同期化している) は、ログイン時に、DCE 証明書とローカル・アクセスを取得するように構成された、スタンザを持つことができます。そのスタンザはおそらく次のようになります。

```
joe:
     SYSTEM = "(DCE AND compat) OR  (DCE[UNAVAIL] AND compat)"
```

これは、**joe** は、DCE 認証と **compat** 認証の両方をパスするか、または DCE が選択不可のために DCE 認証を失敗し、システムに正常にログオンするためには、 **compat** 認証をパスする必要があることを示します。

- もう 1 つの例として、放浪の DCE ユーザー **bob** は、デフォルト・スタンザに以下のものが含まれている、(bob がローカルに定義されていない) システムへのログオンを試行できます。

```
default:
      SYSTEM = "DCE OR (DCE [UNAVAIL]  AND compat)"
```

これは、デフォルト・ユーザーは DCE 認証をパスするか、あるいは DCE が選択不可のために DCE 認証を失敗し、システムへのアクセス権を獲得するためには **compat** 認証をパスする必要があることを示します。もちろん、ユーザーが放浪のユーザーであり、 **/etc/passwd** にローカル・エントリーを持たない場合には、そのユーザーは **compat** をパスできず、DCE が選択不可の場合はすべてのアクセスを否認されます。

- ローカル・ユーザー **joe** がレジストリー内の同じ名前を持つユーザーと同期化しないことが分かっている場合には、ローカル・ユーザー **joe** のスタンザには次のエントリーが含まれています。

```
joe:
      SYSTEM = compat
      registry = files
```

NIS ユーザー **jane** のスタンザには次のエントリーが含まれています。

```
jane:
       SYSTEM = compat
       registry = NIS
```

その目的は、その属性から DCE を明白に除外することにあります。 **joe** または **jane** が、マシンに対し放浪である場合には、ログオンできる唯一の方法は、**compat** (ローカルまたは NIS) 認証が成功するパスワードを用意することです。

- ユーザー **jane** が DCE をとおしてローカル・システムにアクセスする場合には、 jane の **AUTHSTATE** 変数は **DCE** に設定されます。 jane があるファイルに **ls -l** コマンドを実行すると、 UID と GID を名前にマップするために DCE レジストリーが照会されます。 ID が DCE レジストリーに存在しないと、 **ID 照会**オペレーションは **compat** にフォールバックし、ローカル **/etc/passwd** ファイルと **/etc/group** ファイルが照会されます。優先的に照会されるデータベースと、アクセスされるファイル・システム (DFS またはローカル) によっては、ID がミス・マップされることもあることをユーザーは知っている必要があります。

セル間の考慮事項 - セキュリティー統合

AIX/DCE セキュリティー統合を使用することにより、DCE ユーザーが外部セルからログインし、外部セル DCE 証明書を取得することが可能となります。これは、ローカル・セル・レジストリー内にシャドー・プリンシパルを作成することにより実現できます。シャドー・プリンシパルとは、新規の拡張レジストリー属性が付加されたプリンシパルです。この ERA には、外部ユーザーのアカウントが定義されているセルを表す、完全に識別された DCE セル名が入っています。次に、ホーム・セル ERA 値は、ログイン時に外部セルに連絡し、ユーザーを認証するために、セキュリティー統合によって使用されます。シャドー・プリンシパルは、ホーム・セル ERA で指示された外部セルへ認証をリダイレクトするポインターとして働きます。

ユーザーは、シャドー・プリンシパルの名前を指定することによってログインし、ホーム・セル ERA でポイントされた外部セル・アカウントのパスワードを与えます。それによりユーザーは認証され、成功した場合は、外部セル・ユーザーの DCE 証明書を受け取ります。次の例では、ユーザー **henry** は、セル **wildwest.austin.ibm.com** で定義されたアカウントを持っていますが、セル **windsor.austin.ibm.com** に構成されたホストにログインしようとしています。シャドー・プリンシパルである **windsor.austin.ibm.com** セルは、次のように定義されています。

```
Name:           henry
Full Name:      Henry D. Shadow
UNIX ID:        112
UUID:           00000070-e63f-21cf-88-02608c2f5cec
Home Cell ERA:  /.../wildwest.austin.ibm.com
```

シャドー・ユーザー **henry** によるログインが成功すると、このログイン・ユーザーがセル **wildwest.austin.ibm.com** 内のユーザー **henry** の DCE 証明書を持つことになります。

ホーム・セル ERA には、グローバル・セル名と外部セル・ユーザーのユーザー名の両方を含んだ、完全に識別された DCE ユーザ名が入っていることもあります。それにより、シャドー・プリンシパルは、別の名前を持つ外部セル・ユーザーを表すことができるようになります。前記の例でいえば、ホーム・セル ERA が **/.../wildwest.austin.ibm.com/joe** であり、 **henry** によるログインが成功すると、セル **wildwest.austin.ibm.com** 内のユーザー **joe** の DCE 証明書が取得されます。

セル間の管理 - セキュリティー統合

ローカル AIX ユーザーとグループの情報が、セキュリティー統合でのログイン時に DCE レジストリーからリトリーブされます。このため、管理者は、セル間環境におけるセキュリティー統合での使用のために、 DCE ユーザーおよびグループの構成と管理の方法について理解する必要があります。

シャドー・プリンシパルに対する UNIX ID

外部セル・ユーザーのホーム・セル ERA を持つ以外に、シャドー・プリンシパルは、ログイン・ユーザーのローカル UNIX ID を取得するためにも使用されます。それにより、ローカル・セルの ID が外部セルの ID と競合するとき、ユーザー ID の再マップが可能になります。シャドー・プリンシパルは、ホーム・セル ERA によりポイントされる外部セル・ユーザーと同じユーザ ID を持つ必要はありません。ユーザーはログインして、シャドー・プリンシパルのユーザ ID を受け取ります。

アカウント情報

アカウントがシャドー・プリンシパルのためにローカル・セル内に作成される (シャドー・アカウント) と、それはローカル AIX アカウント情報の取得のために使用されます。アカウント情報としては、ユーザーの 1 次グループ、グループ・セット、ログイン・シェル、ホーム・ディレクトリー、および GECOS 情報などがあります。それ以外の場合は、シャドー・プリンシパルのアカウント情報は、ホーム・セル ERA を使用して、外部セルから直接リトリーブされます。ローカル・セルの管理者が外部セル・レジストリーを信用する場合には、ローカル・シャドー・アカウントを作成する必要はありません。

シャドー・プリンシパルがシャドー・アカウントを持っていても、セキュリティー統合を使用してログインするときは、ホーム・セル ERA によりポイントされる外部セル・アカウントをとおして認証されます。

シャドー・グループ

ユーザー・グループ・セットを外部セル・アカウントから取得する場合 (たとえば、ローカル・セル内にシャドー・アカウントを持たないとき) には、ユーザー・グループ・セット内のすべてのグループについて、ローカル・セル・レジストリー内にシャドー・グループを作成する必要があります。シャドー・グループはシャドー・プリンシパルと類似しており、グループが実際に定義されている外部セルをポ

イントする、ホーム・セル ERA を持っています。シャドー・グループ自体は、メンバーを持っていません。また、シャドー・グループは、ホーム・セル ERA によりポイントされる外部グループと同じグループ ID を持っていなければなりません。

ユーザーがローカル・シャドー・アカウントを持っている場合は、そのグループ・セットはローカル・アカウントから取得します。グループはローカル・セル内に存在しなければなりません。

セル間での UNIX ID の管理

セキュリティー統合をセル間環境において拡張して使用する場合には、可能なかぎり、セル間のユーザーおよびグループの UNIX ID はセグメント化することをお勧めします。それぞれのセルは、セルに固有な特定の範囲内にのみ、 DCE のユーザー ID とグループ ID を割り当てます。それにより、シャドー・グループとシャドー・ユーザーの管理がより簡単になります。シャドーは、任意の外部セル内で、そのホーム・セル内と同じ ID を付けて作成することができます (その ID がすでに割り当て済みであることを気にする必要はありません)。

セル間セキュリティー統合を設定するためには次の手順を使用します。

1. DCE セルの間の信頼関係を確立する。

   セル間環境で AIX/DCE セキュリティー統合を使用するためには、影響のあるセルの間に信頼関係を設定する必要があります。

2. セル間フラグにより dceunixd デーモンを始動する。

   dceunixd デーモンがホーム・セル ERA を認識し、セル間操作のために外部セルに連絡することができるように、 dceunixd デーモンは **-i 1** フラグで始動する必要があります。 dceunix に指定されたオプションに **-i 1** フラグを追加します。デーモンがすでに実行中の場合は、それを停止して、再始動します。

3. ホーム・セル拡張レジストリー属性に対しスキーマを定義する。

   シャドー・プリンシパルまたはグループにホーム・セル ERA のインスタンスを付加するためには、その前に、ERA スキーマをレジストリーに定義する必要があります。これはセル内で 1 回だけ、しかもシャドーを作成するセル内でのみ行う必要があります。次に示す dcecp コマンドがホーム・セル ERA を作成します。次の例で指定されている値をそのまま使って、コマンドを入力する必要があります。そのコマンドを実行するためには、セル内に **cell_admin** 特権を持つ必要があります。

```
xattrschema create /.:/sec/xattrschema/shadow_home_cell -attribute {
        {aclmgr
                {{principal r m m m}
                  {group r m m m}}}
        {annotation {ERA for intercell integrated security} }
        {applydefs no}
        {encoding printstring}
        {intercell accept}
        {multivalued no}
        {reserved no}
        {trigtype none}
        {unique no}
        {uuid 35c6cba-e3d2-11cf-b772-268c2f5cce}
}
```

shadow_home_cell をスキーマを作成するときは、この uuid: ″**35c6cba-e3d2-11cf-b772-268c2f5cce**″ を使用する必要があります。

4. ローカル・セルにシャドー・プリンシパルを作成する。

   シャドー・プリンシパルは、通常の DCE プリンシパルに似ていますが、外部セル・ユーザーをポイントする、それに付加されたホーム・セル ERA のインスタンスを持っています。シャドー・プリンシパルが作成された後、 dcecp コマンドを使用して、ホーム・セル ERA を付加することができます。

   ```
   dcecp> principal modify name -add "shadow_home_cell /.../cellname"
   ```

5. ローカル・セルにシャドー・グループを作成する。

   シャドー・プリンシパルについてのアカウント情報を、その外部セル・アカウントから取得する場合は、ユーザー・グループ・セット内のグループに対してシャドー・グループを作成します。シャドー・グループは、グループが実際に定義されている外部セルをポイントする、それに付加されたホーム・セル ERA のインスタンスを持っています。 dcecp コマンドを使用して、ホーム・セル ERA を付加することができます。

   ```
   dcecp> group modify name -add "shadow_home_cell /.../cellname"
   ```

   シャドー・グループは、ホーム・セル ERA によりポイントされる外部グループと同じグループ ID で作成する必要があります。

6. ローカル・セルにシャドー・アカウントを作成する。

   アカウントがローカル・セル内に作成されない場合は、シャドー・プリンシパルについてのアカウント情報は外部セル・アカウントから取り出されます。セルの管理者が外部セル・レジストリーのアカウント情報を信用する場合には、シャドー・アカウントを作成する必要はありません。

7. 統合セキュリティー操作を構成する。

   統合セキュリティー操作のためのシステムの構成方法に関する詳細については、518ページの『手順 - セキュリティー統合』を参照してください。

スリムなクライアント構成におけるセキュリティー統合

スリムなクライアント構成上でセキュリティー統合機能を使用するためには、dceunixd デーモンは **-s** フラグで始動する必要があります。このフラグなしで dceunixd デーモンをスリムなクライアント構成上で始動しようとすると、次のようなエラーが起こります。

```
(dceunixd) Ensure that the valid machine credentials exist and that
the KRB5CCNAME environment variable is not set.
```

dced はスリムなクライアント構成上では稼働しないので、証明サービスは使用できません。 **-s** フラグで dceunixd を始動すると、

ユーザー識別の妥当性検査は行われますが、その証明は行われません。

passwd_override 機能と **group_override** 機能は、スリムなクライアント構成のシステムでは使用できません。

複数の Dceunixd デーモンを持つセキュリティー統合

最高 5 つまでの dceunixd デーモンを **-n numdaemons** フラグで始動できます。デフォルト (**-n numdaemons** フラグなし) は 1 つの dceunixd デーモンです。複数の dceunixd デーモンにより、多数のユーザーを持つシステムのためのセキュリティー統合のキャパシティーを増大することができます。

現在のリリースでは、複数の dceunixd デーモンは、プロセスによってモニターされません。すなわち、何らかの理由で dceunixd を削除する必要がある場合は、dceunixd **-n numdaemons** フラグですでに始動済みの、すべての dceunixd デーモンを管理者が手作業で削除する必要があります。

-n numdaemons フラグと **-d level** フラグは、相互に排他的です。

第42章 接続可能な認証モジュール (Solaris のみ)

接続可能な認証モジュール (PAM) は、複数の認証メカニズムを標準のシステム・プログラムに統合する API およびフレームワークです。 PAM モジュールは、**/usr/lib/security** ディレクトリーに入っている共用ライブラリーです。 PAM ライブラリー・ファイルは、インストールと構成手順の最中に、自動的に **/usr/lib** ディレクトリーにインストールされます。こうした共用ライブラリーには、**認証**、**アカウント**、および**パスワード**という 3 種類のモジュール・タイプがあります。

注: PAM は、オペレーティング・システムに制限があるため、 Solaris 2.5.1 かそれ以前のバージョンでは使用することができません。 Solaris 2.5.1 またはそれ以前のバージョンで DCE * DFS を実行している場合、 **login.dce** コマンドを使用してください。

sso_cell 構成ファイル

PAM は、その他に名前が付けられたセルがない場合、 **/opt/dcelocal/etc/security** ディレクトリーにある **sso_cell** 構成ファイルを使用して、ユーザーが所属するセルを識別します。このファイルは、ユーザーがデフォルトのセル (**defaultcell** フラグにリスト)、ローカル側のセル (マシンが構成されるセル)、あるいはそれらに代わるセル (**altcells** フラグでリスト) の、どれに所属するかを識別するためにも使用されます。

注: **sso_cell** ファイルは、**root** (UID **0**) によって所有され、**0644** モード・ビットの設定 (**-rw-r--r--**) でなければなりません。ファイルがこれら 2 つの要件を満たしていない場合、このファイルは無視され、警告が記録されます。

sso_cell ファイルにデフォルトのセルを定義するには、 **config.dce pam** を使用します。 **defaultcell** フラグは、次のようにセットされます。

```
defaultcell = /.../your.cell.name
```

unconfig.dce pam が、**sso_cell** ファイルを削除します。

sso_cell ファイルを使って 1 つまたは複数の代替セルを定義するには、 **altcells** フラグを次のようにセットします。

```
altcells = /.../alt.cell.name,
/.../another.cell.name
```

sso_cell 構成ファイルを編集すると、ブランク行、スペース、およびコメントを受け入れることができます。

認証モジュール

認証モジュールは、ユーザーのパスワードを検査し、ユーザーの認証ファイルと DFS 処理認証グループ (PAG) を作成します。また、パスワードの有効期限について任意で警告を出します。これは、以下の **pam.conf** ファイル構成オプションを受け入れます。

- **use_first_pass** オプションは、以前のモジュール専用に提供されたパスワードを使って認証を試行することができます。このオプションを使用すると、認証モジュールが最初に呼び出されたときにだけ、パスワード・プロンプトが使用可能になります。
- **try_first_pass** オプションは、連鎖内にある以前のモジュールに提供されたパスワードを使って認証を試行することができます。認証の試行が失敗するか、以前のモジュールがない場合には、 DCE パスワードを要求するプロンプトが出されます。
- **ignore_root** オプションは、 root という名前のユーザーか UNIX UID がゼロのユーザーの認証をすべて試行できなくします。
- **nowarn** オプションは、パスワードの有効期限に関する警告を含め、警告を生成できなくします。
- **warnexpire=***N* オプション (ここで、*N* は正の整数) は、ユーザー・パスワードの有効期限が次の *N* 日内に切れる場合に、警告メッセージを発行します。デフォルト設定は、30 日です。
- **bind_pe_site option** は、**/opt/dcelocal/etc/security/pe_site** ファイルだけを使って、レジストリー・サーバーを配置するために PAM を使用できるようにします。このオプションを設定するのと、**BIND_PE_SITE** 環境変数を設定するのは同じことです。
- **proxiable** フラグは、現在のチケットとは異なるネットワーク・アドレスを持つチケットを、**認証**モジュールからユーザーに発行できるようにします。プロキシー可能チケットが認可されている場合、ユーザーでプロキシー可能チケットが認可されるように、レジストリーにあるユーザー・アカウントを指定する必要があります。
- **forwardable** フラグは、現在の TGT とは異なるネットワーク・アドレスを持つチケット認可チケット (TGT) を、プリンシパルに発行できるようにします。転送可能なチケットが認可されている場合、プリンシパルで転送可能チケットが認可されるように、レジストリーにあるユーザー・アカウントを指定する必要があります。

お勧めする認証モジュールの構成は、**try_first_pass ignore_root** です。

アカウント・モジュール

account モジュールは、**getpwnam** と DCE レジストリーを任意で検査して、ユーザー UNIX UID に対する双方の承諾を確認し、ユーザーのログイン・コンテキスト認証を検査します。また、ユーザーの DCE hostdata に対する正規のアクセス権を確認します。お勧めする **login** プログラムの構成は、 **ignore_root require_match require_cert nondce_ok** です。

注: アカウント・モジュールを指定すると、**passwd_override** ファイルは無視されます。

- **require_match** オプションは、システム **getpwnam_r** 関数 (**/etc/passwd** ファイルの情報を検査) を介して UNIX UID を取得するユーザーへのログオン・アクセスを使用できなくします。このオプションは、DCE ディレクトリーに保存された UNIX UID とはマッチングしません。

- **require_cert** オプションは、DCE ログイン・コンテキストが認証されない場合、 DCE へアクセスできないようにします。認証モジュールの実行時に、有効な UID がゼロの場合、 DCE PAM によって取得するログイン・コンテキストが認証されます。システムによるログイン処理 (**login**、**telnet**、**dtlogin**、またはローカル側の **root** として実行する任意の PAM アプリケーションなど) の **require_cert** を指定します。
- **ignore_root** オプションは、 **root** という名前のユーザーか UNIX UID がゼロのユーザーの認証をすべて試行できなくします。
- **nowarn** オプションは、パスワードの有効期限に関する警告を含め、警告を生成できなくします。
- **nondce_ok** オプションは、DCE ログイン・コンテキストを持たず、 DCE ディレクトリーに存在しないユーザーが、DCE にアクセスできるようにします。ユーザーが DCE ディレクトリーに存在していて、ログイン・コンテキストを持たない場合には、アクセスは拒否されます。
- **check_acl** オプションは、 UUID 14d5e798-2f75-11d1-9c40-9e62070caa7 を指定したローカル・ホスト用の DCE hostdata オブジェクトの ACL に対する読み取りアクセスを承認し、 DCE アクセスを認可します。
- **alt_acl=***UUID* オプション (ここで、 *UUID* は任意のローカルの hostdata オブジェクトの UUID) は、任意の代替 DCE hostdata オブジェクト上の ACL を検査します。このオプションを **check_acl** オプションの代替オプションとして、または **check_acl** オプションと組み合わせるオプションとして使い、 ACL 検査またはデュアル ACL 検査をそれぞれ行ってください。同じプロシージャーを使用して、**login**、**ftp**、 **dtlogin** を介したコンソール・アクセス、および **telnet** などの様々な ACL を設定します。 **check_acl** オプションの代わりに **alt_acl** を設定することにより、他の UUID を指定した同じようなオブジェクトを検査することもできます。
- **bind_pe_site** は、**/opt/dcelocal/etc/security/pe_site** ファイルだけを使って、レジストリー・サーバーを配置するために PAM を使用できるようにします。このオプションを設定するのと、**BIND_PE_SITE** 環境変数を設定するのは同じことです。

PAM を使ってログインの ACL を検査するには、次のようにします。

1. 次のコマンドを使って、**dced hostdata** インターフェースを使用できるようにします。

   ```
   $dced -x
   ```

   注: **dced hostdata** を使用できるようにするには、 **-x** オプションを使用する必要があります。 **dced hostdata** が使用できない場合、または hostdata オブジェクトが存在しない場合、アクセスは拒否されます。
2. 以下のコマンドを使用して、特定の UUID を指定したホスト上に **hostdata** オブジェクトを作成します。

   ```
   dcecp -c hostdata create /.:/hosts/machine/config/hostdata/login_svc¥
   -storage /opt/dcelocal/var/security/login_svc -data { {} }
   -uuid 14d5e798-2f75-11d1-9c40-9e62070caa7
   -annotation {The ACL on this object determines login permissions for this
   host.}
   ```
3. **pam.conf** ファイルにある **pam_dce** アカウント行に **check_acl** 引き数を追加します。

注: r 許可により、ログインの許可が決まります。

パスワード・モジュール

password モジュールは、DCE レジストリーにあるユーザーのパスワードを変更し、 DCE パスワードの効力に関するポリシーを緩やかに実行します。また、パスワードが変更されるとき、ユーザーの DCE 認証を任意でリフレッシュします。

それぞれのモジュール・タイプは、サービスに応じて **/etc/pam.conf** ファイルにリストされます。たとえば、サービスには **login**、**ftp**、および **dtlogin** があります。モジュールは **pam.conf** ファイルに表示される順番で読み取られるため、それぞれのタイプおよびサービスごとに複数のモジュールをスタックまたはリストすることができます。必要に応じて、任意で、または十分なモジュールを複数指定することもできます。

PAM を使用するには、PAM 構成ファイル **/etc/pam.conf** を編集して、 DCE ライブラリーを指定する行を挿入する必要があります。この行は、UNIX ライブラリーを指定する同じような行のすぐ後に置きます。以下に示すのは、**pam.conf** ファイルの例です。

```
# Authentication management#
login   auth optional   /usr/lib/security/pam_dce.so.1 try_first_pass ignore_root
rlogin  auth optional   /usr/lib/security/pam_dce.so.1 try_first_pass ignore_root
dtlogin auth optional   /usr/lib/security/pam_dce.so.1 try_first_pass ignore_root
other   auth optional   /usr/lib/security/pam_dce.so.1 try_first_pass ignore_root
# Account management
login   account optional /usr/lib/security/pam_dce.so.1 ignore_root require_match
dtlogin account optional /usr/lib/security/pam_dce.so.1 ignore_root require_match
other   account optional /usr/lib/security/pam_dce.so.1 ignore_root require_match
# Password management
other   password optional /usr/lib/security/pam_dce.so.1 refresh
```

DCE アプリケーションが PAM API を呼び出して、 PAM ライブラリー **libpam.so.1** にリンクすると、 PAM フレームワークは該当するモジュールを検出してロードし、そのモジュールが提供する PAM サーバー・プロバイダー・インターフェース (SPI) を呼び出します。

接続可能な認証モジュールが DCE 提供のライブラリーを使用するよう構成される場合、移行処理時にシステムをリブートする前に、**/etc/pam.conf** ファイルを変更し、 DCE ライブラリーへの参照があれば削除する必要があります。 **dcesetup upgrade_uninstall** コマンドを使用すると、 **/etc/pam/.conf** の DCE ライブラリーへの参照が削除されます。変更をしない場合、PAM はアップグレード・アンインストール時に削除した DCE ライブラリーを見つけようとするため、リブート後にシステムへログインできません。

移行後に PAM を使用する場合、DCE ライブラリーへの参照を削除する前に、**/etc/pam.conf** のバックアップ・コピーを作成できます。その後、**dcesetup upgrade_install** の処理を完了したら復元します。

次に示すのは、Solaris 7 上で構成され、 UNIX ライブラリーだけを参照する、デフォルトの **/etc/pam.conf** ファイルの例です。

```
#
# PAM configuration
#
# Authentication management
```

```
#
login   auth required   /usr/lib/security/pam_unix.so.1
login   auth required   /usr/lib/security/pam_dial_auth.so.1
#
rlogin  auth sufficient /usr/lib/security/pam_rhosts_auth.so.1
rlogin  auth required   /usr/lib/security/pam_unix.so.1
#
dtlogin auth required   /usr/lib/security/pam_unix.so.1
#
rsh     auth required   /usr/lib/security/pam_rhosts_auth.so.1
other   auth required   /usr/lib/security/pam_unix.so.1
#
# Account management
#
login   account required      /usr/lib/security/pam_unix.so.1
dtlogin account required      /usr/lib/security/pam_unix.so.1
#
other   account required      /usr/lib/security/pam_unix.so.1
#
# Session management
#
Other   session required      /usr/lib/security/pam_unix.so.1
#
# Password management
#
Other   password required    /usr/lib/security/pam_unix.so.1
```

/etc/pam.conf にある DCE ライブラリー参照の削除を忘れて、システムをリブートすると、ログインできません。この場合、単一ユーザー・モードでシステムをリブートできます。 **/etc/pam.conf** のバックアップ・コピーを保管します。 **etc/pam.conf** から DCE ライブラリーへの参照を削除します。もう一度、通常モードでリブートします。ログインして残りの移行を完了します。その後、 **/etc/pam.conf** のバックアップ・コピーを復元し、 PAM 機能をもう一度使用可能にします。

第43章 レジストリー・オブジェクトへのアクセス

この章では、レジストリーの中のオブジェクトに適用される許可について説明します。認可はレジストリー・データベースの構成方法に基づいて与えられるので、この章では最初に、レジストリー・データベースの構造について簡単に説明します。次に、レジストリー・データベース、レジストリー ACL マネージャー、および初期レジストリー ACL の中のそれぞれのオブジェクトの許可について説明します。

dcecp コマンドと **acl_edit** コマンドはともに、レジストリー・オブジェクトの ACL エントリーの作成、修正、および削除の機能を持っています。それぞれが ACL エントリーに対して実行する操作の説明については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* のそれぞれのコマンドを参照してください。

レジストリー・データベース

注: DCE セキュリティー・レジストリーおよび LDAP 統合機能に関するこのトピックについては、 *IBM DCE for AIX and Solaris バージョン 3.2: DCE セキュリティー・レジストリーと LDAP 統合* を参照してください。

レジストリーは、次の主要なディレクトリーに構造化されます。
- **principal** ディレクトリー - プリンシパルについての情報を含む
- **group** ディレクトリー - グループについての情報を含む
- **org** ディレクトリー - オーガニゼーションについての情報を含む

ディレクトリーの他に、レジストリーには、**policy** オブジェクト、 **replist** オブジェクト、および **xattrschema** オブジェクトが含まれています。これらのオブジェクトはすべて、マシン構成時にレジストリーが作成されるときに作成されます。**policy** オブジェクトには、レジストリーの特性とポリシー、およびオーガニゼーション・ポリシーに適用される情報が含まれており、 **replist** オブジェクトには、DCE セルの中のレプリカについての情報が含まれており、 **xattrschema** オブジェクトには、拡張レジストリー属性 (ERA) についての情報が含まれています。**policy** 情報と **replica** 情報は、 **dcecp registry** コマンドを使用して、いつでも変更することができます。 **xattrschema** オブジェクトは、 **dcecp xattrschema** コマンドを使用して変更します。

principal、**group**、または **org** のいずれかのディレクトリーに単純オブジェクトを作成するとき、必要に応じてサブディレクトリーが作成されます。たとえば、**preludes/villa/lobos** のようなプリンシパルを追加する場合、サブディレクトリー **preludes** と **villa** が作成されます。データの編成を支援するために、これらのサブディレクトリーを使用することができます。サブディレクトリー内のすべてのオブジェクトを削除すると、サブディレクトリー自体も削除されます。 (**principal**、**group**、または **org** の各ディレクトリーは削除できません。)

レジストリー内のオブジェクトに与えられる認可は、オブジェクトがレジストリー・データベースのどの構造に入るかによって異なります。
534ページの図54 は、レジストリー・データベースを図示しています。四角形はコンテナー・オブジェクト (ディレクトリー) を表しています。楕円は単純オブジェク

トを表しています。 図54 は、トップレベルの **principal**、**group**、および **org** の各ディレクトリーのみを示しています。レジストリーは、ユーザーがサブディレクトリーを作成すれば、サブディレクトリーを持つことができます。

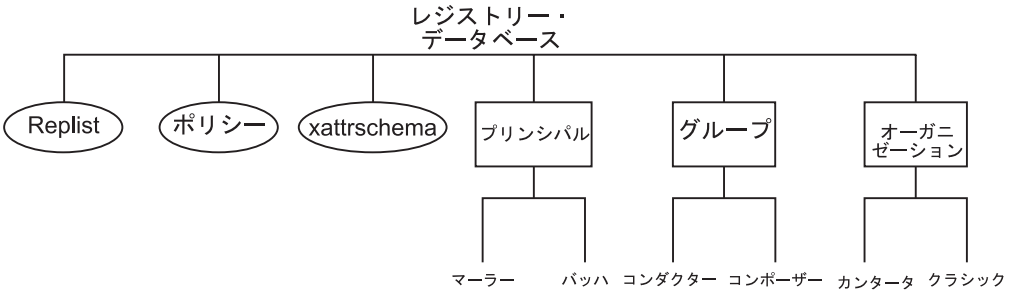


図 54. レジストリー・データベースの構造

レジストリーの許可

表30 は、レジストリーに存在するオブジェクト・タイプに対して与えることができる、認可をリストしています。

表 30. レジストリー・オブジェクトの許可

許可	意味
A	レプリカを操作するコマンドを実行する (**sec_admin**)。
a	認証情報を修正する。
c	オブジェクトに関する ACL を変更する。すべてのレジストリー ACL に、 **c** (コントロール) 許可を指定するエントリーが 1 つなければなりません。
d	オブジェクトの内容から削除する。
D	レジストリーからオブジェクトを削除する。
f	プリンシパル、グループ、またはオーガニゼーションのフルネームを変更する。
g	プリンシパルをグループに追加する。
i	オブジェクトの内容に追加する。
m	管理情報を変更する。
M	このグループまたはオーガニゼーションにメンバーを追加、またはメンバーを削除する。メンバーをグループに追加するためには、追加されるプリンシパルに対する **g** 許可も必要です。
n	ディレクトリー、プリンシパル、グループ、またはオーガニゼーションの名前を変更する。
u	ユーザー情報を変更する。
r	管理、認証、およびユーザーの各情報を表示する。
t	指定されたプリンシパルについて、グループまたはオーガニゼーションの各メンバーシップをテストする。

管理情報、認証情報、およびユーザー情報

レジストリーは、レジストリーの中のオブジェクトに関する 3 種類の異なる情報、すなわち、管理情報、認証情報、およびユーザー情報を含んでいます。それぞれのオブジェクト・タイプごとに保持される特定の情報について、以下のサブセクションで要約します。

管理情報

管理情報には次のものがあります。

- レジストリーのポリシーと属性について
  - アカウント存続期間
  - パスワード最小長
  - パスワード存続期間
  - パスワードにスペースを含めることができるかどうか
  - パスワードを非英数字のみで構成できるかどうか
  - パスワード有効期限
  - 最小チケット存続時間
  - デフォルト・チケット存続時間
  - プリンシパル、グループ、またはオーガニゼーションの作成時に自動的に与えられる、最小 UNIX ID を定義する数値
  - プリンシパル、グループ、またはオーガニゼーションの作成時に UNIX ID として (自動的または手動で) 与えられる、最大数を定義する数値
  - 暗号化したパスワードを表示するかどうか (シャドー・パスワード属性)
- プリンシパルについて
  - アカウント、グループ、およびオーガニゼーションの名前
  - プリンシパルのフルネームを示すテキスト・ストリング
  - プリンシパルのオブジェクト作成割り当て量
  - プリンシパルが 1 次名を別名に、そして別名を 1 次名に変更できるかどうか
  - プリンシパルのユーザー ID (UID)
  - プリンシパルの固有のユーザー ID (UUID)
  - プリンシパルのアカウントの有効期限
  - プリンシパルのアカウントのアカウント有効フラグ
  - アカウントが、クライアントまたはサーバーとして作動できるプリンシパルに対するものかどうかを示すフラグ
- グループについて
  - グループの 1 次名
  - グループのフルネームを示すテキスト・ストリング
  - グループの 1 次名を別名に、そして別名を 1 次名に変更できるかどうか
  - グループのグループ ID (GID)
  - プロジェクト・リスト包含属性
  - グループの UUID
- オーガニゼーションについて
  - オーガニゼーションの 1 次名
  - オーガニゼーションの 1 次名を別名に、そして別名を 1 次名に変更できるかどうか
  - オーガニゼーションのフルネームを示すテキスト・ストリング
  - オーガニゼーションのオーガニゼーション ID (ORGID)
  - オーガニゼーションの UUID

  - アカウント存続期間
  - パスワード最小長
  - パスワード存続期間
  - パスワード有効期限
  - パスワードにスペースを含めることができるかどうか
  - パスワードを非英数字のみで構成できるかどうか
- **xattrschema** オブジェクトについて
  - xattrschema を変更できるかどうか

認証情報

認証情報には次のものがあります。

- レジストリーのポリシーと属性について
  - 最大チケット存続時間
  - チケットを更新できる最大時間
- プリンシパルについて
  - プリンシパルのアカウントの最大チケット存続時間
  - プリンシパルのアカウントに対して発行されたチケットを更新できる最大時間
  - プリンシパルのアカウントが最後に変更された日時 (Good Since Date)
  - プリンシパルのアカウントが使用可能にされた日時 (Last Changed Date)
  - プリンシパルのアカウントの作成者と作成日
  - アカウント使用の説明
  - プリンシパルのアカウントが、先日付チケット、転送可能チケット、更新可能チケット、または代用可能チケットを発行できるかどうか
  - DCE 認証サービスが、チケット発行チケット認証に基づいて、プリンシパルのアカウントに対してチケットを発行できるかどうか、またはプリンシパルがサービスのためのチケットを直接入手する必要があるかどうか。
  - プリンシパルのアカウントに重複セッション・キーを発行できるかどうか

ユーザー情報

ユーザー情報としては、プリンシパルのアカウントに関する次の情報があります。

- パスワード
- ホーム・ディレクトリー
- 各種の情報 (GECOS 情報)
- ログイン・シェル
- パスワード有効フラグ

プリンシパル、グループ、またはオーガニゼーションの作成に必要な許可

図55 は、プリンシパル、グループ、またはオーガニゼーションの作成に必要な許可を示しています。

図 55. プリンシパル、グループ、またはオーガニゼーションの作成に必要な許可

プリンシパル、グループ、またはオーガニゼーションを作成するためには、作成したプリンシパル、グループ、またはオーガニゼーションが入るディレクトリーに関する **i** 許可が必要です。たとえば、プリンシパル **preludes/villa/lobos** を作成するためには、 **villa** に対する **i** 許可を持っている必要があります。

プリンシパル、グループ、またはオーガニゼーションの削除に必要な許可

図56 は、プリンシパル、グループ、またはオーガニゼーションの削除に必要な許可を示しています。

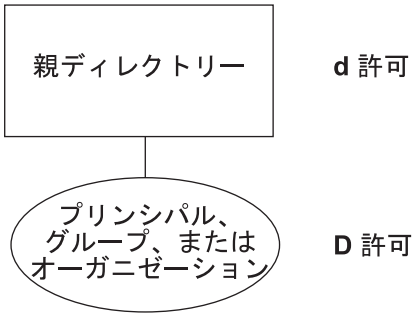


図 56. プリンシパル、グループ、またはオーガニゼーションの削除に必要な許可

プリンシパル、グループ、またはオーガニゼーションを削除するためには、次の許可が必要です。

- 削除するプリンシパルが存在するディレクトリーに関する **d** 許可
- 削除するプリンシパル、グループ、またはオーガニゼーションに関する **rD** 許可

たとえば、プリンシパル **preludes/villa/lobos** を削除するためには、ディレクトリー **preludes/villa** に対する **d** 許可と、プリンシパル **preludes/villa/lobos** に対する **rD** 許可を持っていなければなりません。

アカウントの追加に必要な許可

user create コマンドを使用して、アカウントを追加するときに、プリンシパルがまだグループまたはオーガニゼーション (あるいはその両方) のメンバーでない場合、 **dcecp** コマンドは、アカウントで指定されたグループまたはオーガニゼーションにプリンシパルを追加します。このため、アカウントの追加に必要な許可には、グループまたはオーガニゼーションにメンバーを追加するために必要な許可が含まれることがあります。次のトピックは、アカウントの追加に必要な許可の説明の中で述べられています。

- アカウントを追加すると同時に、アカウントで指定されるグループまたはオーガニゼーションのメンバーとして、プリンシパルを追加するために必要な許可。 (『アカウントの追加に必要な許可』を参照。)
- すでにプリンシパルが、指定されたグループとオーガニゼーションのメンバーになっている、アカウントを追加するために必要な許可。 (538ページの『プリンシパルが、すでにグループとオーガニゼーションのメンバーであるアカウントの追加』を参照。)

- アカウントを追加し、アカウントで指定されたグループにのみ (プリンシパルがすでにオーガニゼーションのメンバーになっているため) プリンシパルを追加するために必要な許可。 (539ページの『アカウントとプリンシパルのグループのみへの追加』を参照)
- アカウントを追加し、アカウントで指定されたオーガニゼーションにのみ (プリンシパルがすでにグループのメンバーになっているため) プリンシパルを追加するために必要な許可。 (540ページの『アカウントとプリンシパルのオーガニゼーションのみへの追加』を参照)

アカウントとアカウント・プリンシパルのグループおよびオーガニゼーションへの追加

図57 は、アカウントおよびアカウント・プリンシパルをグループまたはオーガニゼーションに追加するために必要な許可を示しています。

図57. アカウントとアカウント・プリンシパルをグループとオーガニゼーションに追加するために必要な許可

アカウントを追加して、アカウントで指定されたグループとオーガニゼーションにアカウントのプリンシパルを自動的に追加するためには、次の許可が必要です。
- アカウントのプリンシパルに関する **maug** 許可
- アカウントで指定されたグループに関する **tM** 許可
- アカウントで指定されたオーガニゼーションに関する **rtM** 許可
- レジストリー **policy** オブジェクトに関する **r** 許可

たとえば、グループ **composers** とオーガニゼーション **pianists** に関連した、プリンシパル **preludes/villa/lobos** のアカウントを作成するためには、次の許可が必要です。
- **preludes/villa/lobos** に関する **maug** 許可
- グループ **composers** に関する **tM** 許可
- オーガニゼーション **pianists** に関する **rtM** 許可
- レジストリー **policy** オブジェクトに関する **r** 許可

プリンシパルが、すでにグループとオーガニゼーションのメンバーであるアカウントの追加

539ページの図58 は、プリンシパルがすでにグループまたはオーガニゼーションのメンバーであるアカウントを追加するために必要な許可を示しています。

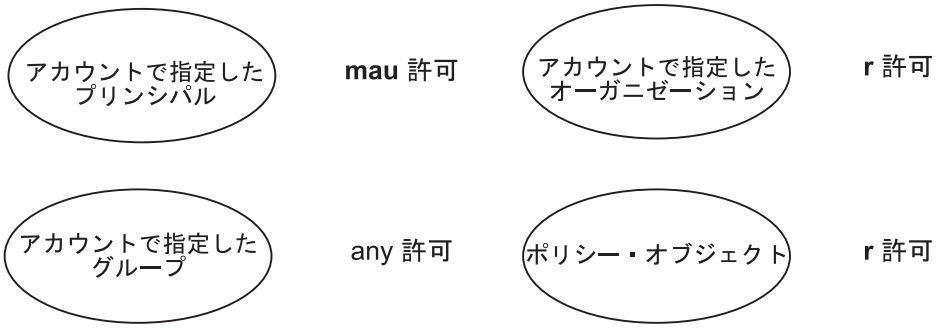


図 58. すでにプリンシパルがグループおよびオーガニゼーションのメンバーであるアカウントの追加

アカウントのプリンシパルを追加する必要のないアカウントを、アカウントで指定されたグループおよびオーガニゼーションに追加するためには、次の許可が必要です。

- アカウントのプリンシパルに関する **mau** 許可
- アカウントで指定されたグループに関する任意の種類の少なくとも 1 つの許可
- アカウントで指定されたオーガニゼーションに関する **r** 許可
- レジストリー **policy** オブジェクトに関する **r** 許可

たとえば、グループ **composers** とオーガニゼーション **pianists** に関連した、プリンシパル **preludes/villa/lobos** のアカウントを作成するためには、次の許可が必要です。

- **preludes/villa/lobos** に関する **mau** 許可
- グループ **composers** に関する任意の種類の少なくとも 1 つの許可
- オーガニゼーション **pianists** に関する **r** 許可
- レジストリー **policy** オブジェクトに関する **r** 許可

アカウントとプリンシパルのグループのみへの追加

図59 は、アカウントとプリンシパルをグループにのみ追加するために必要な許可を示しています。

図 59. アカウントとプリンシパルをグループにのみ追加するための許可

アカウントを追加して、そのアカウントのプリンシパルをグループに追加するためには (プリンシパルはすでに、アカウントで指定されたオーガニゼーションのメンバーになっている)、次の許可が必要です。

- アカウントのプリンシパルに関する **maug** 許可

- アカウントで指定されたグループに関する **tM** 許可
- アカウントで指定されたオーガニゼーションに関する **r** 許可
- レジストリー **policy** オブジェクトに関する **r** 許可

アカウントとプリンシパルのオーガニゼーションのみへの追加

図60は、アカウントとプリンシパルをオーガニゼーションにのみ追加するために必要な許可を示しています。

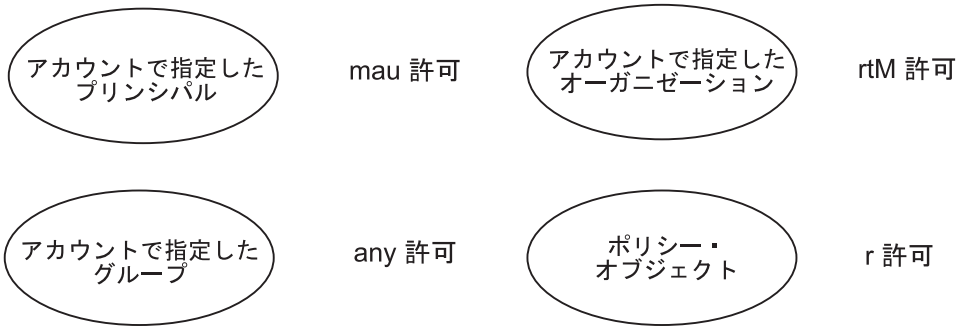


図 60. アカウントとプリンシパルをオーガニゼーションにのみ追加するための許可

アカウントを追加して、そのアカウントのプリンシパルをオーガニゼーションに追加する (プリンシパルがすでに、アカウントで指定されたグループのメンバーになっている) ためには、次の許可が必要です。

- アカウントのプリンシパルに関する **mau** 許可
- アカウントで指定されたグループに関する任意の種類の少なくとも 1 つの許可
- アカウントで指定されたオーガニゼーションに関する **rtM** 許可
- レジストリー **policy** オブジェクトに関する **r** 許可

アカウントの削除のために必要な許可

図61 は、アカウントの削除に必要な許可を示しています。

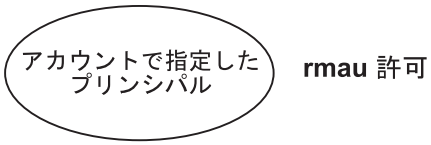


図 61. アカウントを削除するために必要な許可

アカウントを削除するためには、アカウントで指定されたプリンシパルに関する **rmau** 許可が必要です。たとえば、**preludes/villa/lobos** という名前のプリンシパルのアカウントを追加または削除するためには、 **preludes/villa/lobos** に関する **rmau** 許可が必要です。

メンバーをグループに追加するために必要な許可

541ページの図62 は、メンバーをグループに追加するために必要な許可を示しています。

図 62. メンバーをグループに追加するために必要な許可

メンバーをグループに追加するには、以下の許可が必要です。
- プリンシパルを追加する先のグループに関する **rM** 許可
- 追加するプリンシパルに関する **rg** 許可

たとえば、プリンシパル **preludes/villa/lobos** をグループ **composers** に追加するためには、次の許可が必要です。
- グループ **composers** に関する **rM** 許可
- プリンシパル **lobos** に関する **rg** 許可

メンバーをオーガニゼーションに追加するために必要な許可

図63 は、メンバーをオーガニゼーションに追加するために必要な許可を示しています。

図 63. メンバーをオーガニゼーションに追加するために必要な許可

メンバーをオーガニゼーションに追加するためには、次の許可が必要です。
- プリンシパルを追加する先のオーガニゼーションに関する **rM** 許可
- 追加するプリンシパルに関する **r** 許可

たとえば、プリンシパル **preludes/villa/lobos** をオーガニゼーション **pianists** に追加するためには、次の許可が必要です。
- オーガニゼーション **pianists** に関する **rM** 許可
- プリンシパル **lobos** に関する **r** 許可

グループまたはオーガニゼーションからメンバーを削除するための許可

図64 は、グループまたはオーガニゼーションからメンバーを削除するために必要な許可を示しています。

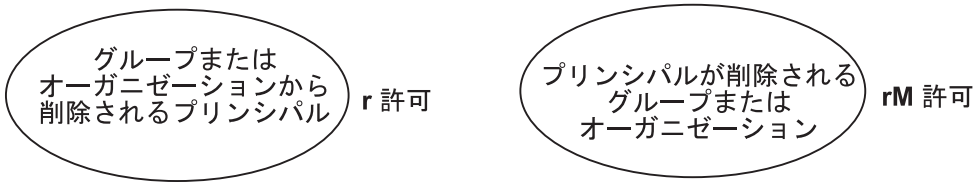


図 64. グループまたはオーガニゼーションからメンバーを削除するための許可

グループまたはオーガニゼーションからメンバーを削除するためには、プリンシパルを削除するグループまたはオーガニゼーションに関する **rM** 許可、および削除するプリンシパルに関する **r** 許可が必要です。

たとえば、グループ **composers** からプリンシパル **preludes/villa/lobos** を削除するためには、次の許可が必要です。
- グループ **composers** に関する **rM** 許可
- プリンシパル **lobos** に関する **r** 許可

プリンシパル、グループ、またはオーガニゼーションの各フルネームを変更するために必要な許可

図65 は、プリンシパル、グループ、またはオーガニゼーションの各フルネームを変更するために必要な許可を示しています。

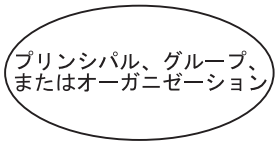


図 65. プリンシパル、グループ、またはオーガニゼーションの各フルネームを変更するために必要な許可

プリンシパル、グループ、またはオーガニゼーションのフルネームを変更するためには、変更するプリンシパル、グループ、またはオーガニゼーションに関する **rf** 許可が必要です。

プリンシパル、グループ、またはオーガニゼーションの管理情報の変更に必要な許可

図66 は、プリンシパル、グループ、またはオーガニゼーションについての管理情報の変更に必要な許可を示しています。

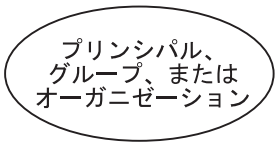


図 66. プリンシパル、グループ、またはオーガニゼーションの管理情報を変更するために必要な許可

プリンシパル、グループ、またはオーガニゼーションの管理情報を変更するためには、管理情報を変更するオブジェクトに関する **rm** 許可が必要です。

アカウントの管理情報、認証情報、およびユーザー情報 (パスワードを除く) の変更に必要な許可

543ページの図67 は、アカウントの管理情報、認証情報、およびユーザー情報 (パスワードを除く) の変更に必要な許可を示しています。

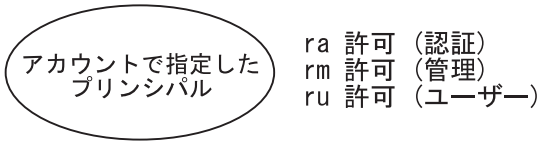


図 67. アカウントの管理情報、認証情報、およびユーザー情報 (パスワードを除く) の変更に必要な許可

アカウントの管理情報、認証情報、およびユーザー情報 (パスワードを除く) のすべてを変更するためには、アカウントで指定されたプリンシパルに対する次の許可が必要です。

- 認証情報を変更するための **ra** 許可
- 管理情報を変更するための **rm** 許可
- ユーザー情報を変更するための **ru** 許可

アカウントのパスワード変更に必要な許可

図68 は、アカウントのパスワードの変更に必要な許可を示しています。

図 68. アカウントのパスワードの変更に必要な許可

アカウントのパスワードを変更するためには、アカウントで指定されたプリンシパルに対する次の許可が必要です。

- アカウントのプリンシパルに関する **ru** 許可
- レジストリー **policy** オブジェクトに関する **r** 許可

レジストリー・ポリシーと属性の認証情報と管理情報の変更に必要な許可

図69 は、レジストリー・ポリシーと属性の認証情報と管理情報の変更に必要な許可を示しています。

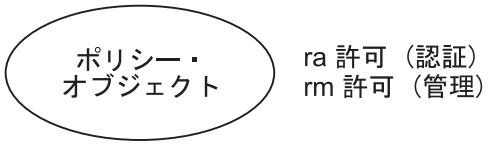


図 69. レジストリーのポリシーと属性の管理情報と認証情報を変更するために必要な許可

dcecp registry modify コマンドを使用してレジストリーの管理情報または認証情報を変更するためには、レジストリー **policy** オブジェクトについて、認証情報の変更には **ra** 許可、管理情報の変更には **rm** 許可が必要です。

レプリカを操作するコマンドの実行に必要な許可

図70 は、レプリカを操作するコマンドの実行に必要な許可を示しています。

図 70. レプリカを操作するコマンドの実行に必要な許可

レプリカを操作するコマンドを実行するためには、 **replist** オブジェクトに関する次の許可が必要です。

- レプリカ情報を表示するコマンドを除く、すべてのコマンドを実行するための **A** 許可。レプリカ情報を表示するコマンドは、 **replist** オブジェクトに関する許可を必要としません。
- レプリカを削除するコマンドを実行するための **d** 許可。

拡張レジストリー属性型の作成に必要な許可

図71 は、 ERA タイプの作成に必要な許可を示しています。

図 71. 拡張レジストリー属性型を作成するために必要な許可

レジストリー・スキーマに ERA タイプを作成するためには、 **xattrschema** オブジェクトに関する **i** 許可が必要です。

拡張レジストリー属性型の削除に必要な許可

図72 は、ERA タイプの削除に必要な許可を示しています。

図 72. 拡張レジストリー属性型を削除するために必要な許可

ERA タイプを削除するためには、 **xattrschema** オブジェクトに関する **d** 許可が必要です。

拡張レジストリー属性型の表示に必要な許可

図73 は、レジストリーのスキーマ・データベース内の、 1 つまたは複数の ERA を表示 (**dcecp xattrschema show** コマンドによって) に必要な許可を示しています。

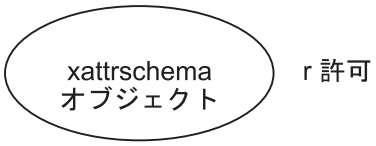


図 73. 拡張レジストリー属性の表示に必要な許可

ERA タイプを表示するためには、 **xattrschema** オブジェクトに関する **r** 許可が必要です。

拡張レジストリー属性型の変更に必要な許可

図74 は、ERA タイプの変更に必要な許可を示しています。

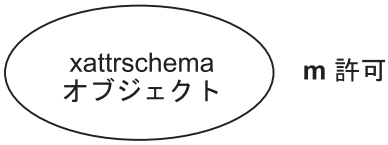


図 74. 拡張レジストリー属性型の変更に必要な許可

ERA タイプを変更するためには、 **xattrschema** オブジェクトに関する **m** 許可が必要です。

レジストリー・オブジェクトの ACL の変更に必要な許可

図75 は、レジストリー・オブジェクトの ACL の変更に必要な許可を示しています。

図 75. レジストリー・オブジェクトの ACL の変更に必要な許可

レジストリー・オブジェクトの ACL を変更するためには、変更する ACL を持つオブジェクトに関する **c** 許可が必要です。レジストリー・オブジェクトには、**policy** オブジェクト、すなわち、プリンシパル、グループ、またはオーガニゼーションがあります。

スレーブ・レプリカが必要とする許可

スレーブ・レプリカを正しく初期設定して機能させるためには、スレーブ・レプリカは、**replist** オブジェクト (**/.:/sec/replist**) に対する **i** 許可、 **m** 許可、および **l**

許可を持っていなければなりません。スレーブ・サーバーは、それが作動するマシンの識別の下で稼働します。マシン名は、次の形式のローカル・ホスト・プリンシパル名です。

host/*hostname***/self**

必要な ACL エントリーが追加されるのは、 **dce_config** ツールが DCE セルのセキュリティー・サーバーを最初に構成し、ユーザーが新規のスレーブ・レプリカを作成するためにこのツールを使用するときです。エントリーの形式は次のとおりです。

user:host/*hostname***/self:imI**

レジストリー ACL マネージャー

レジストリー ACL マネージャーは、5 つのマネージャー・タイプで構成されます。これらのマネージャー・タイプは、レジストリー内の 5 つのタイプのオブジェクトが必要とする、各種の ACL 意味体系を処理するために使用されます。たとえば、プリンシパル ACL マネージャー・タイプは、レジストリー内のすべての **principal** オブジェクトの ACL を制御します。 **group** オブジェクトは、**principal** オブジェクトの場合と異なる一連の許可を必要とするため、 **group** オブジェクトの ACL を制御する、別個のグループ ACL マネージャー・タイプがあります。

それぞれの ACL マネージャーに対して、すべての許可または ACL エントリー・タイプが有効であるわけではありません。 表31 は、各 ACL マネージャーごとに、有効または無効な許可、および無効な ACL エントリー・タイプを要約しています。

表 31. ACL マネージャーと有効な許可および ACL エントリー・タイプ

マネージャー・タイプ	制御対象	有効な許可	無効な ACL エントリー・タイプ
dir	**directory** オブジェクト	**rcidDn**	**user_obj, group_obj**
policy	**policy** オブジェクト	**rcma**	**user_obj, group_obj**
principal	**principal** オブジェクト	**rcDnfmaug**	**group_obj**
group	**group** オブジェクト	**rctDnfmM**	**user_obj**
org	**org** オブジェクト	**rctDnfmM**	**user_obj, group_obj**
replist	レプリカ・リスト	**cidmIA**	**user_obj, group_obj**
xattrschema	ERA タイプ	**rcidm**	**user_obj, group_obj**

初期レジストリー ACL

レジストリー・データベースを作成するときには、**principal**、**group**、および **org** の各ディレクトリーと、**policy**、**replist**、および **xattrschema** の各オブジェクトには、初期 ACL が与えられます。レジストリーの中に新しいオブジェクトを作成するとき、それらのオブジェクトは、**principal**、**group**、および **org** の各ディレクトリーから ACL を継承します。キーが必要な初期 ACL エントリーの ACL エント

リー・キーは、レジストリー・データベースを作成するプリンシパルの名前 (レジストリー作成者として **sec_create_db** コマンドに与える)、または名前が指定されない場合には **root** です。 (**sec_create_db** およびレジストリー作成者に関する詳細については、 475ページの『第38章 レジストリーの設定』を参照してください。)

レジストリー・データベースの作成時に作成される初期 ACL について、次のリストで説明します。このリストでは、*rgy_creator* が、レジストリー作成者として指定されたプリンシパルを示しています。

注: プラットフォームの構成ツールは、これらの初期 ACL を更新することができます。

- **principal** オブジェクトの場合

```
unauthenticated:r--------
user_obj:r---f--ug
user:rgy_creator:rcDnFmaug
other_obj:r-------g
any_other:r--------
```

- **group** オブジェクトの場合

```
unauthenticated:r-t-----
user:rgy_creator:rctDnfmM
group_obj:r-t-----
other_obj:r-t-----
any_other:r-t-----
```

- **org** オブジェクトの場合

```
unauthenticated:r-t-----
user:rgy_creator:rctDnfmM
other_obj:r-t-----
any_other:r-t-----
```

- **policy** オブジェクトの場合

```
unauthenticated:r----
user:rgy_creator:rcma
other_obj:r----
any_other:r----
```

- **directory** オブジェクトの場合

```
unauthenticated:r-----
user:rgy_creator:rcidDn
other_obj:r-----
any_other:r-----
```

- **replist** オブジェクトの場合

```
user:cell_admin:cidmA-
```

- **xattrschema** オブジェクトについて

```
unauthenticated:r-----
user:cell_admin:rcidm
other_obj:r-----
any_other:r-----
```

第44章 DCE 監査サービス

監査は、分散システムで重要な役割を果たします。分散アプリケーションでクリティカル・イベントの検出と記録を行うためには、適切な監査機能が必要になります。

DCE のキー・コンポーネントである監査機能は、DCE 監査サービスによって提供されます。この章では、DCE 監査サービスについて紹介します。

DCE 監査サービスの機能

DCE 監査サービスには、次の機能があります。

- 監査デーモン (**auditd**) は、指定の基準に基づいて監査レコードのロギングを行います。
- アプリケーション・プログラム・インターフェース (API) は、監査イベントを記録するためのアプリケーション・サーバー・プログラムの一部として使用することができます。これらの API は、監査レコードおよび監査証跡の分析を行うことができる、ツールを作成するためにも使用することができます。
- 監査デーモンへの管理コマンド・インターフェースは、特定の基準に基づいて記録しようとするイベントの選択においてデーモンに指示します。このインターフェースは、 DCE 制御プログラム (**dcecp**) または監査制御 API をとおしてアクセスされます。
- イベント分類メカニズムを使用すると、1 組のイベントを論理的にグループ化することができ、管理が容易になります。
- 監査レコードは、ログまたはコンソールに出力することができます。

DCE 監査サービスのコンポーネント

DCE 監査サービスは、次の基本コンポーネントを持っています。

- アプリケーション・プログラミング・インターフェース (API)

  アプリケーション・サーバーがクライアントにサービスするときにクリティカル・イベントの検出と記録を行うために使用する、機能を提供します。アプリケーション・プログラマーは、監査イベントの記録を行うために、アプリケーション・サーバー・プログラムの特定のコード・ポイント で、これらの機能を使用します。その他の API は、監査イベント・レコードの調査と分析を行うツールを作成するために、使用することができます。
- 監査デーモン

  監査デーモンは次のサービスを提供します。
  - フィルターと中央監査証跡ファイルを保守する。
  - DCE 制御プログラム (**dcecp**) が監査デーモンを制御するために使用する、RPC インターフェースをエクスポートする。

DCE 監査サービスの概念

このセクションでは、DCE 監査サービスの管理に関連した概念のいくつかについて説明します。

監査クライアント

すべての RPC ベースのサーバー、すなわち、DCE サーバーとユーザー作成のアプリケーションは、潜在的な監査クライアントです。 DCE セキュリティー・サービスと分散タイム・サービスを監査することができます。つまり、コード・ポイント (次のセクションで説明します) が、すでにこれらのサービスの適切な場所に置かれています。

注: 監査サブシステムの主要な性質のため、 RPC サーバーとしては機能しない RPC クライアントから監査イベントを生成することはお勧めできません。サーバー・サイドにおける監査では、監査レコードが記録される場所の数も制限され、以下の理由から監査証跡からの情報収集もさらに容易になります。すなわち、一般的なクライアント・サーバー・アプリケーションにおいては、サーバーがクライアントよりはるかに少ない傾向があるからです。

しかし、クライアント・サイドの監査は、クライアントがフィルター (環境に設定された **DCEAUDITFILTERON** で始動される) を使用しようとしないかぎり、機能します。 RPC サーバーではない RPC クライアントは、フィルターの状況がいつ変更になるかを知る方法がなく、その変更を反映するように適応できないため、フィルターを使用できません。これは、監査サブシステムの機能要件にとってクリティカルな機能です。

監査デーモンは、自身を監査することもできます。

中央監査証跡ファイルを使用するためには、監査クライアントは監査デーモン・オブジェクトに対する **log** 許可を持っていなければなりません。監査デーモンに対する許可については、 563ページの『第45章 DCE 監査サービスの管理用タスク』 で説明します。

コード・ポイント

コード・ポイントとは、アプリケーション・サーバー・プログラム内の、 DCE 監査 API が使用される場所です。コード・ポイントは一般的に、監査を必要とするアプリケーション・サーバーによって提供される、オペレーションまたは機能と対応しています。たとえば、バンク・サーバーが現金引き出し機能 **acct_withdraw( )** を提供する場合、この機能は監査可能イベントと見なすことができ、コード・ポイントとして指定できます。

コード・ポイントは、すでに DCE セキュリティー・サービス、分散タイム・サービス、および監査サービス・コードの中の適切な場所に置かれています。 DCE セキュリティー・サービスのコード・ポイントとその関連イベントについては、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* に記述されています。 DCE 分散タイム・サービスのコード・ポイントと関連イベントについては、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレン*

ス に記述されています。 DCE 監査サービスのコード・ポイントと関連イベントについては、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* に記述されています。

監査イベント

監査イベントとは、監査クライアントが記録したいイベントのことです。一般的に、監査イベントにはシステムの保全性が要求されます。たとえば、クライアントが銀行口座から現金を引き出すと、これは銀行口座のセキュリティー違反を伴う可能性があるため、監査イベントになる可能性があります。

監査イベントは、アプリケーション・サーバー・コードのコード・ポイントと関連しています。

イベント番号

どの監査イベントにも、アプリケーション・プログラマーによってイベント番号が割り当てられます。イベント番号は、0xC0000000 などの 32 ビットの整数です。イベント番号については、 *IBM DCE Version 3.2 for AIX and Solaris: Application Development Guide--Core Components* でより詳細に説明されています。

イベント・クラス

監査イベントは、イベント・クラスに論理グループ化することができます。イベント・クラスは、何組かのイベントを 1 つの値で指定することを可能にする、効率のよいメカニズムを提供します。一般的に、イベント・クラスは、何らかの共通性を持った監査イベントでもって構成されます。たとえば、銀行サーバー・プログラムでは、現金取引 (預金、引き出し、および振替) をイベント・クラスにグループ化することができます。イベント・クラスについては、563ページの『第45章 DCE 監査サービスの管理用タスク』でも説明します。

イベント・クラス・ファイル

イベント・クラスは、*イベント・クラス・ファイル* 内で定義されます。すべてのイベント・クラス・ファイルは、 *dcelocal*/**etc/audit/ec** ディレクトリー内に作成する必要があります。

監査可能イベントを分類するために、デフォルトのイベント・クラス・ファイルが DCE セキュリティー・サービス、タイム・サービス、および監査サービスから提供されます。これらのファイルは、このいずれかのサービスがインストールされる時に、ホスト・システムにインストールされます。

イベント・クラスの名前はそのファイル名と同じです。各イベント・クラスは、イベント・クラス・ファイル内で定義されます。

イベント・クラス・ファイルからイベント番号を削除または追加するか、あるいは新しいイベント・クラス・ファイルを作成することによって、新しいイベント・クラスを定義することができます。

イベント・クラス名

各イベント・クラスは、それぞれに割り当てられた記号名を持っています。次に、ベンダーが従う必要がある、イベント・クラスの望ましい名前の形式を示します。

ec_*org_product_class*

ここで、

org イベント・クラスを定義するオーガニゼーションまたは企業の名前です。

product
イベント・クラスが定義されている製品の名前です。

class イベント・クラスの特徴です。

以下に、イベント・クラス名の例を 2 つ示します。

- **ec_osf_dce_authentication** - OSF の DCE コア・コンポーネントの認証イベント・クラスを定義します。
- **ec_transarc_encina_update** - Transarc の Encina® の更新イベント・クラスを定義します。

ユーザー独自の監査要件に適合するようにイベント・クラスを定義することもできます。次に、これらのイベント・クラスの望ましい名前の形式を示します。

dce_*server-name_class*

ここで、*class* はイベント・クラスの特徴です。

イベント・クラス番号

独自のイベント・クラスを定義する場合には、それをイベント・クラス番号と関連付ける必要があります。イベント・クラス番号は 32 ビットの整数で、ホストで定義されているそれぞれのイベント・クラスに固有のものである必要があります。各イベント・クラス番号は、 セット *ID* とクラス・イベント *ID* で構成されるタプルです。セット ID はイベント・クラスのセットに対応しており、これは OSF によってオーガニゼーションまたはベンダーに割り当てられます。クラス・イベント ID は、そのイベント・クラスのセット内の 1 つのイベント・クラスを識別します。オーガニゼーションまたはベンダーは、イベント・クラス番号を生成するために、クラス・イベント ID 番号の発行を管理します。

イベント・クラス番号の構造と管理は、IP アドレスの構造と管理にたとえることができます。 IP アドレスは、ネットワーク ID (セット ID に類似する) とホスト ID (クラス ID に類似する) のタプルであることを思い出してください。

イベント・クラス番号のフォーマット

イベント・クラス番号は、オーガニゼーション内のイベント・クラスの数によって、 5 つのフォーマット (A から E) のいずれかに従います。イベント・クラス番号のフォーマットは、その 4 つの高位ビットから判別することができます。

フォーマット A は、クラス・イベント ID に 16 ビット以上を必要とする、大きなオーガニゼーション (OSF や主要な DCE ベンダーなど) で使用されます。このフォーマットでは、 7 ビットがセット ID に、24 ビットがクラス・イベント ID に割り振られます。セット ID としてゼロ (0) を指定したフォーマット A イベント・クラス番号は、 OSF に割り当てられます。つまり、OSF が使用するすべてのイベント・クラス番号では、最有効バイトにゼロが指定されています。

フォーマット B は、クラス・イベント ID に 8 〜 16 ビットを必要とする、中規模サイズのオーガニゼーションで使用されます。

フォーマット C は、クラス・イベント ID に 8 ビットより少ないビットを必要とする、小規模のオーガニゼーションで使用されます。

フォーマット D は OSF によって管理されておらず、セル内で自由に使用することができます。このイベント・クラス番号はセル間で固有にすることができないため、複数のセルにインストールされているアプリケーション・サーバーでは使用しないでください。

フォーマット E は、将来の使用のために予約されています。

最有効ビットに 110 を指定した番号 (つまり 0xC0000000 ～ 0xDFFFFFFF) は、セル内のローカル使用のために予約されています。

イベント・クラス番号のフォーマットが、図76 に図示されています。 (クラス・イベント ID には 『event-id』 というラベルが付いています)。

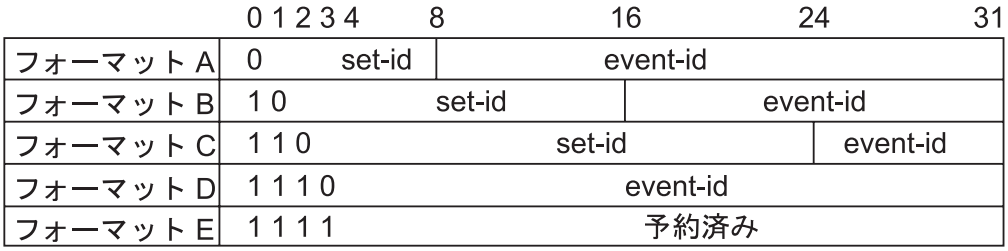


図 76. イベント・クラス番号のフォーマット

セル管理者は、ローカル・イベント・クラス番号とその名前の管理と割り当てに責任を負います。

フィルター

コード・ポイントが識別されてアプリケーション・サーバーに配置されると、コード・ポイントに対応するすべての監査イベントが、これらの監査イベントの結果に関係なく、1 つの監査証跡ファイルに記録されます。しかし、すべての条件の下ですべての監査イベントを記録することは実用的でもなく、必要でもありません。フィルターは、特定の条件が満たされた場合にのみ監査レコードを記録する手段です。管理者は、DCE 制御プログラムを使用してフィルターを定義することができます。

フィルターは、条件を指定するフィルター・ガイドから成り立っています。フィルター・ガイドは、条件 (結果) が満足された場合に取るべき処置も指定します。

フィルターでは次の質問に答えていきます。

- 誰を監査するか
- どのイベントを監査するか
- 監査レコードを書き込むためには、これらのイベントの結果がどのようでなければならないか
- 監査レコードは、監査証跡ファイルに記録されるか、システム・コンソールに表示されるか、あるいはその両方か。

たとえば、銀行サーバー・プログラムの場合、監査レコードを書き込むための条件を下記のようにすることができます。

『DCE セル内のすべての顧客 (監査するユーザー) が行い、アクセス否認 (イベントの結果) のため失敗した、すべての引き出しトランザクション (監査イベント) に関する監査レコードを記録する。』

フィルター・サブジェクト識別

フィルターは 1 つのフィルター・サブジェクトと関連付けられます。このフィルター・サブジェクトは、フィルターの適用先 を指示します。フィルター・サブジェクトは、イベントを生じさせた分散アプリケーションのクライアントです。フィルター・サブジェクトには、 2 つの部分、すなわち、フィルター・タイプとキーがあります。

8 つのフィルター・タイプがあります。

- **principal** - ローカル・セル内の DCE プリンシパル。
- **foreign_principal**- 外部セル内の DCE プリンシパル。
- **group** - ローカル・セル内の DCE グループ。
- **foreign_group** - 外部セル内の DCE グループ。
- **cell** - ネットワーク内の DCE セル。
- **cell_overridable** - ネットワーク内の DCE セル。このタイプは、より特定したフィルター・タイプにより指定変更されることがあります。
- **world** - 分散アプリケーションのすべてのクライアント。
- **world_overridable** - 分散アプリケーションのすべてのクライアント。このタイプは、より特定したフィルター・タイプにより指定変更されることがあります。

キーは、**principal**、**foreign_principal**、**group**、 **foreign_group**、**cell**、および **cell_overridable** の各フィルター・タイプの特定の名前です。 **world** および **world_overridable** フィルター・タイプにはキーはありません。

フィルター・ガイド

フィルターには、1 つまたは複数のガイドが含まれています。フィルター・ガイドには、監査条件、監査アクション、およびイベント・クラスの 3 つのエレメントが含まれています。

監査条件は、監査レコードが監査証跡に書き込まれるために必要とされる、イベントの結果 (1 つまたは複数) を指定します。これらの結果は互いに排他的ではありません。監査条件は次のとおりです。

- **success** - イベントが成功したときのみ記録する。
- **failure** - イベントが失敗したときのみ記録する。
- **denial** - アクセス否認のためイベントが失敗したときのみ記録する。

監査アクションは、監査レコードが書き込まれる場所を指定します。 監査アクションは、次のとおりです。

- **alarm** - システム・コンソールに監査レコードを表示する。
- **log** - 監査デーモンをとおすか、または直接、監査証跡ファイルに監査レコードを記録する。

- **ems** - 適切に登録されている ENS コンシューマーにイベントを渡す ENS サーバーにイベントを指定する。

これらの監査アクションは互いに排他的ではありません。

フィルター・ガイドの **all** アクションには、 **log** および **alarm** アクションに加えて、 **ems** アクションが含まれるようになりました。前のリリースの DCE で監査フィルターを作成し、DCE 3.2 へ移行するときに、 **all** アクションを指定する場合、このアクションには **log** および **alarm** アクションが含まれますが、 **ems** アクションは含まれません。

監査フィルターを作成して変更するときには、何らかの DCE コア・サービス (たとえば、セキュリティーや cds) の監査イベントを含むフィルター・ガイドに、**ems** アクションを指定しないでください。指定すると、監査フィルター操作を使用する場合に、 DCE コア・サービスがハングする可能性があります。

4 番目のアクション (そのアクションが使用するときだけ意味がある) は **none** です。このアクションは、特定のプリンシパル、グループまたはセルにオーバーライド可能なフィルターをオーバーライドするのに使用します。

フィルター・ガイドの 3 番目の要素は、 (特定のフィルター・サブジェクト識別について) フィルターが適用されるイベント・クラス (1 つまたは複数) を指定します。

フィルター・ガイドの例

次は、2 つのガイドのあるフィルターの例です。

```
{principal_cell_admin}
{{dce_sec_authent dce_sec_server} success log}
{dce_sec_authent {failure denial} {log alarm}}
```

これは、プリンシパル・フィルター (キーは **cell_admin**) です。ガイド 1 は、ユーザーが **cell_admin** で、アクションが成功した場合に、イベント・クラス **dce_sec_authent** (DCE セキュリティー認証に関係あるイベント) または **dce_sec_server** (DCE セキュリティー・レプリカに関係ある認証) のすべてのイベントについて、監査レコードが記録されるように指定します。ガイド 2 は、ユーザー **cell_admin** の **dce_sec_authent** にあるイベントで、結果が `access denied` または `unspecified failure` であるすべてのイベントが記録され、コンソールに表示されるように指定します。

フィルター規則

フィルター規則は、別のフィルターのオーバーラップするガイドを解決するために使用されます。 2 つのフィルター規則、すなわち、オーバーライド規則と最高水準点規則があります。

オーバーライド規則の下では、オーバーライド可能フィルター (つまり、**cell_overridable** タイプと **world_overridable** タイプ) は、より具体的なフィルターによって無効にされます。オーバーライド規則は、相補フィルターを認めるメカニズムとして使用されます。 principal または group 用のフィルターは、 cell または world 用のフィルターよりもより具体的なフィルターです。

最高水準点規則は、オーバーライド規則の後で適用されます。複数のフィルターをクライアントに適用できる場合、それらのフィルターで指定されるアクション (log または alarm) の共用体が適用されます。

クライアントのプリンシパル、グループ、またはセル識別がフィルターのキーと一致する場合には、フィルターはクライアントに適用可能です。 **world** フィルターと **world_overridable** フィルターにはキーはなく、すべてのクライアントに適用できます。クライアントに適用可能なフィルターが複数ある場合には、それらのフィルターで指定されるアクション (**log**、**alarm**、または **ems**) の共用体が採用されます。

フィルター規則の使用例

オーバーライド可能フィルターの使用については、次のシナリオで説明します。

企業 (セル) X の Alice は、ある操作 (イベント・クラス **critical_transactions**) を活動化する責任者です。その企業のその他のプリンシパルは、同じ操作を活動化することを認められていますが、それは、たとえば、Alice が使えない時など、特定の条件のもとに限られます。システム管理者は、イベント結果 (つまり audit conditions = all)、またはこの操作を活動化する人に関係なく、監査レコードを記録することを希望しています。活動化する人が Alice でない場合に、警報を生成することも希望しています。この指定は、次の 2 つのフィルターによって実現されます。

```
Filter 1:
{principal Alice}
{critical_transactions all log}
Filter 2:
{cell_overridable X}
{critical_transactions all {log alarm}}
```

Alice が **critical_transactions** イベント・クラスでイベントを呼び出すとき、プリンシパル・フィルター (フィルター 1) が適用できます。その理由は、そのキーが Alice の識別に一致するためです。プリンシパル・フィルターは、セル・フィルターよりもさらに具体的です。セル・フィルター (フィルター 2) も、 Alice (Alice はセル X に属している) に適用されますが、セル・フィルターがオーバーライド可能なため、プリンシパル・フィルターによってオーバーライドされます。企業 (セル) X のその他のプリンシパルの場合、適用可能なフィルターはセル・フィルター (フィルター 2) だけです。このように、これらの同じイベントにより、監査レコードが記録されることになり、警報が出されます。

オーバーライド不能のワールド・フィルターとセル・フィルターも便利です。たとえば、それらが使用できない場合は、管理者は、セル中のフィルターをセル全体に有効にするためには、 1 つのセルのグループとプリンシパルのすべてのフィルターを削除しなければなりません。 (セルがセキュリティー問題の原因と思われるときは、システム管理者は、*一時的* な、オーバーライド不能のセル・フィルターを採用することができます。)

557ページの図77 は、異なるタイプのフィルター間のオーバーライド関係を図示しています。フィルター・タイプ X からフィルター・タイプ Y への矢印は、 X が Y をオーバーライドすることを示しています。

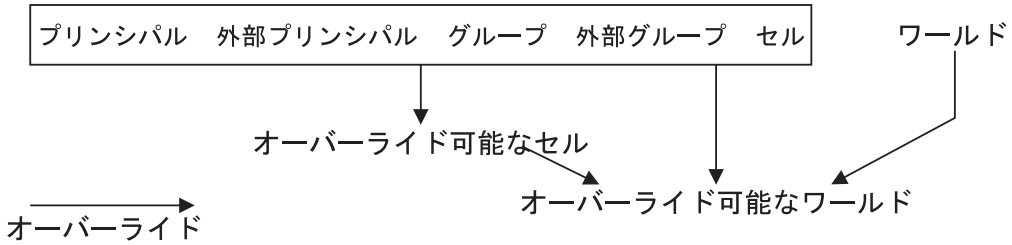


図 77. フィルター・タイプ間のオーバーライド関係

DCE グループは一般的に、アクセス認可を付与する目的で定義されます。グループ・フィルターは、*グループに属するプリンシパルの監査* を指定するのではなく、*グループの特権を使用する意図の監査* を指定します。すなわち、プリンシパルがグループの特権を (PAC にグループを組み込むことによって) 使用する意図がある場合を除いて、グループ・フィルターは、そのグループのメンバー・プリンシパルに監査の影響を及ぼしません。グループ・フィルターは、グループの特権を使用する意図を監査するように定義されているため、他のフィルターから独立しており、オーバーライド不能です。

監査証跡ファイル

監査証跡ファイルには、監査デーモンによって書き込まれる、すべての監査レコードが含まれます。*中央監査証跡ファイル* または*ローカル監査証跡ファイル* のいずれかを指定できます。

中央監査証跡ファイルは、監査デーモンの始動時に、監査デーモンによって作成されます。 **dce_aud_open( )** 機能で監査証跡ファイルの名前が指定されない場合は、デフォルトにより、すべての監査レコードは監査デーモンに送られ、監査デーモンはそれらのレコードを中央監査証跡ファイルに格納します。

証跡ファイルの名前を指定して **dce_aud_open(f)** 機能を呼び出した場合、この名前がローカル監査証跡ファイルのパス名になり、すべての監査レコードがそのファイルに送られます。

DCE 監査での管理とプログラミング

DCE 監査サービス管理タスクの多くは、アプリケーション・プログラマーが行うタスクと関連付けられます。それらの管理タスクについて理解するためには、 DCE 監査サービスのいくつかのプログラミングの局面について熟知する必要があります。このセクションでは、代表的な DCE 監査サービスのプログラミングと管理シナリオ、およびそれらのタスクについて説明します。

銀行業務サーバーの例でこのシナリオを示します。

プログラマーのタスク

アプリケーション・プログラマーは、アプリケーション・サーバー・プログラムでの監査を可能にするために、DCE 監査 API を使用します。特に、プログラマーは次に示すタスクを行います。

1. アプリケーション・サーバー・プログラムの監査イベントに対応するコード・ポイントを識別する。

   たとえば、銀行業務サーバー・プログラムは、**acct_open( )**、**acct_close( )**、**acct_withdraw( )**、 **acct_deposit( )**、および **acct_transfer( )** の各機能を持つことができます。これらの機能を、それぞれコード・ポイントとして指定することができます。つまり、これらの機能は記録される可能性のある (フィルターによって決まる) 監査イベントであることを意味します。

   ```
   acct_open( )              /* first code point */
   acct_close( )             /* second code point */
   acct_withdraw( )          /* third code point */
   acct_deposit( )           /* fourth code point */
   acct_transfer( )          /* fifth code point */
   ```

2. 各コード・ポイントにイベント番号を割り当てる。イベント番号は、監査証跡をオープンする **dce_aud_open( )** API、およびコード・ポイントの監査レコードを初期設定する **dce_aud_start( )** API によって、パラメーターとして使用される。プログラマーは、それらのイベント番号をサーバーのヘッダー・ファイルの中で定義することができます。

   たとえば、次のとおりです。

   ```
   /* event number for the first code point, acct_open( ) */
   #define evt_vn_bank_server_acct_open      0x01000000
   /* event number for the second code point, acct_close( ) */
   #define evt_vn_bank_server_acct_close     0x01000001
   /* event number for the third code point, acct_withdraw( ) */
   #define evt_vn_bank_server_acct_withdraw  0x01000002
   /* event number for the fourth code point, acct_deposit( ) */
   #define evt_vn_bank_server_acct_deposit   0x01000003
   /* event number for the fifth code point, acct_transfer( ) */
   #define evt_vn_bank_server_acct_transfer  0x01000004
   ```

3. **dce_aud_open( )** API に対する呼び出しを、アプリケーション・サーバーの初期設定ルーチンに追加する。これによって、監査証跡ファイルがオープンされます。この機能は、そのパラメーターの 1 つとして、最下位の番号のイベント、(この場合は、**acct_open( )**) のイベント番号を使用します。たとえば、次のとおりです。

   ```
   main( )
   /* evt_vn_bank_server_acct_open is the lowest event number */
   dce_aud_open(aud_c_trl_open_write, description,
                evt_vn_bank_server_acct_open,
                5, &audit_trail, &status);
   ```

4. 監査イベント・ロギング機能をアプリケーション・サーバー・コードの各コード・ポイントに追加する。それらの機能は、各コード・ポイントで次のことを行います。

   - **dce_aud_start( )** API を使用して、監査レコードを初期設定する。この機能は、イベントを表すコード・ポイントにイベント番号を割り当てます。そのため、この機能はイベント番号をそのパラメーターの 1 つとして使用します。
   - **dce_aud_put_ev_info( )** API を使用して、監査レコードにイベント固有の情報を追加する。
   - **dce_aud_commit( )** API を使用して、監査レコードをコミットする。この機能は、監査レコードを監査証跡ファイルに書き込みます。

   次に、銀行サーバー・プログラムのコード・ポイントでこれらの API を使用する例を示します。 (これは、完全な例ではありません。)

```
acct_open( )     /* first code point */
/* Uses the event number for acct_open( ), */
/*                              evt_vn_bank_server_acct_open */
dce_aud_start(evt_vn_bank_server_acct_open,
              binding,options,outcome,&ard, &status);
 if (ard) /* If events need to be logged */
      dce_aud_put_ev_info(ard,info,&status);
 if (ard) /* If events were logged */
      dce_aud_commit(at,ard,options,format,&outcome,&status);
acct_close( )   /* second code point */
/* Uses the event number for acct_close( ), */
/*                              evt_vn_bank_server_acct_close */
dce_aud_start(evt_vn_bank_server_acct_close,
              binding,options,outcome,&ard, &status);
 if (ard) /* If events need to be logged */
      dce_aud_put_ev_info(ard,info,&status);
 if (ard) /* If events were logged */
      dce_aud_commit(at,ard,options,format,&outcome,&status);
```

5. サーバーのシャットダウン時に、メイン・サーバー・ルーチンで **dce_aud_close( )** API を使用して、監査証跡ファイルをクローズする。たとえば、次のとおりです。

```
dce_aud_close(audit_trail, &status);
```

管理者のタスク

管理者は、イベント・クラス・ファイルとフィルター・ガイドを作成するために、監査クライアント・アプリケーション・サーバー・プログラムの中の、さまざまなコード・ポイントを表すイベント番号を次の方法で使用します。

1. 管理者は、アプリケーション・サーバー・プログラマーから、コード・ポイントのイベント番号 (それぞれの監査イベントを表す) を入手する。この例では、コード・ポイントには次のイベント番号が割り当てられています。

   acct_open( )
   0x01000000
   acct_close( )
   0x01000001
   acct_withdraw( )
   0x01000002
   acct_deposit( )
   0x01000003
   acct_transfer( )
   0x01000004

   (イベント番号はすべて完全な順番になっていなければなりません。つまり、欠番があってはいけません。)

2. 管理者は、2 つのイベント・クラスを作成することを決定する。それらは、**acct_open( )** および **acct_close( )** で構成される **account_creation_operations** クラスと、 **acct_withdraw( )**、**acct_deposit( )**、および **acct_transfer( )** で構成される **account_balance_operations** クラスです。管理者は、イベント・クラス **account_creation_operations** に、イベント・クラス番号 0xC0000006 を割り当てます。イベント・クラス **account_balance_operations** には、イベント・クラス番号 0xC0000007 が割り当てられます。

イベント・クラスを作成するためには、管理者は、 2 つのファイル (各イベント・クラスごとに 1 つずつ) を作成して編集します。これらのファイルのそれぞれの名前は、それぞれが表すイベント・クラスと同じになります。各ファイルには、各イベント・クラス内のイベント番号が含まれます。

名前が **account_creation_operations** のファイルは、次のように編集されます (# (番号記号) で始まる行はコメント行です)。

```
# Event class number of account_creation_operations
ECN = 0xC0000006
# Event number of acct_open( )
0xC1000000
# Event number of acct_close( )
0xC1000001
```

account_balance_operations のファイルは、次のように編集されます。

```
# Event class number of account_balance_operations
ECN = 0xC0000007
# Event number of acct_withdraw( )
0xC1000002
# Event number of acct_deposit( )
0xC1000003
# Event number of acct_transfer( )
0xC1000004
```

管理者は、両方のファイルを *dcelocal*/**etc/audit/ec** ディレクトリーに格納します。

3. 管理者は、次の 2 つのフィルターの作成を決定する。 1 つはセル (セル **/.:/torolabcell** の場合) 内のすべてのユーザー用、もう 1 つは他のすべてのユーザー用です。

   セル内のすべてのユーザーのフィルターのガイドは以下のとおりです。

   - 次の条件に基づき、イベント・クラス **account_balance_operations** 内のイベントのみを監査する。
   - アクセス否認のためにそのイベント・クラスの操作が失敗した場合にのみ、監査レコードを書く。
   - 最初の条件が満たされた場合には、監査証跡ファイルにのみ監査レコードを書く。
   - 管理者は次に、DCE 制御プログラムの **audfilter create** コマンドを使用して、次のフィルターを作成する。

     ```
     dcecp> audfilter create  {cell /.../torolabcell} -attribute ¥
     > {account_balance_operations denial log}
     dcecp>
     ```

   その他のすべてのユーザーのフィルターのガイドは以下のとおりです。

   - 次の条件に基づき、両方のイベント・クラスのイベントを監査する。
   - そのイベント・クラスの操作が成功するか、失敗するか、あるいはアクセス否認のため失敗した場合に、監査レコードを書く。
   - 監査レコードを監査証跡ファイルとコンソールの両方に書く。

   次に、このフィルターを作成するための **dcecp** セッションを示します。

   ```
   dcecp> audfilter create  world -attribute ¥
   > {{account_balance_operations account_creation_operations} {alarm log} all}
   dcecp>
   ```

563ページの『第45章 DCE 監査サービスの管理用タスク』は、 DCE 制御プログラムの **audfilter create** コマンドの詳細について説明しています。

第45章 DCE 監査サービスの管理用タスク

この章では、DCE 監査の使用を開始するための基本ステップについて概説します。さらに、この章では、 DCE 監査サービスについて行われる次の管理用タスクについて説明します。

- DCE 監査環境変数の設定。
- DCE 監査デーモンの開始 (または終了)。
- DCE 監査デーモンへのアクセスの制御。
- 1 組の監査イベントを論理的にグループ化するためのイベント・クラスの作成と保守。イベント・クラスは、イベント・クラス・ファイルを編集することによって作成されます。
- 監査証跡ファイルに監査イベントを記録するための基準を設定する、フィルターの作成と保守。
- DCE 監査デーモンの監査ロギング・サービスの使用可能化と使用不能化。
- DCE 監査デーモンの属性の修正と照会。
- 監査証跡ファイルの制御と表示。
- DCE 保守容易性ルーティング・ファイルの使用。

AIX および Solaris における DCE 監査の使用

次の手順は、DCE セル管理者が迅速かつ容易に AIX および Solaris 上で DCE 監査の使用を開始できるように、支援することを目的としています。この手順は、DCE サーバーで提供されるデフォルト監査イベントを扱っており、ユーザー自身のアプリケーションにおける DCE 監査の使用方法について説明する意図はありません。 DCE セキュリティー・サーバー・デーモンを監査することが主要なフォーカスですが、概念は DCE 監査一般にも適用できます。

auditd デーモンの構成

DCE auditd デーモンを構成するために次のコマンドを実行します。

```
config.dce audit
```

auditd デーモンは、少なくとも、セキュリティー・サーバーを構成するホスト上には構成する必要があります。次の dcecp コマンドを使用して、ローカル監査デーモンの状況を表示することができます。

```
dcecp -c aud show
```

DCE 監査クライアントは、auditd デーモンのサービスを使用して監査を行います。この auditd デーモンは、クライアントから監査イベントの通知を受け、監査レコードを監査証跡に書き込みます。たとえば、dtsd time デーモンと secd security server デーモンは監査クライアントです。

DCE サーバーの停止と再始動

DCE サーバーは、ホスト上で監査サービスを使用できるようにするために、**DCEAUDITON** 環境変数を使用して再始動する必要があります。

```
stop.dce
export DCEAUDITON=1
start.dce
```

Start.dce は、DCE デーモンを起動する前に、内部的に **DCEAUDITON = 1** にセットします。

監査レコードの収集

DCE サーバーは、監査レコードを収集し、監査証跡に格納します。監査サブシステムは、どの監査イベントについて監査レコードを作成し、どれを監査証跡に書き込むべきかを判別するために、フィルターを使用します。デフォルトにより、 auditd は、監査レコードを収集するために、 world フィルター (監査の構成時に作成される) を使用します。

セキュリティー・サーバーの監査証跡は、以下のものに格納されます。

```
/opt/dcelocal/var/security/sec_audit_trail
```

監査証跡の表示

監査を使用する目的は監査証跡を表示することです。監査証跡を調べることにより、どのようなクリティカル・イベントがいつ発生し、誰により実行されたのかが表示できるため、 DCE セルのセキュリティーの改善を助けます。

監査証跡ファイルに対し次の dcecp コマンドを使用することによって、監査証跡の監査レコードを表示します。

```
dcecp> audtrail show /opt/dcelocal/var/security/sec_audit_trail
```

名前に基づく許可を使用したログイン試行の監査と RPC サーバー機能における変更

DCE の従来のバージョンでは、監査済みイベントの許可メカニズムが名前だけに基づいている場合、無効な名前 (たとえば、有効な DCE プリンシパル名でないストリング) を使用すると、イベントが適切に監査されることはありませんでした。

今回のリリースでは、セキュリティーに関して名前だけに基づくイベント (名前に基づいて許可された RPC、事前認証) が監査され、DCE プリンシパル名以外の名前が指定されると、イベントは適切に監査されます。使用される名前が有効な DCE 名であれば、プリンシパルの UUID はクライアント *UUID* として監査レコードに保存されます。指定された名前が有効でない場合には、クライアント UUID はすべてゼロにセットされ、許可状況は **rpc_c_authz_name** (**dce_aud_print( )** および **dcecp audtrail show()** により、*名前を使用した許可* として表示される) になります。また、その名前は最初のイベントに固有の項目として監査レコードに保存されます。

これは名前に基づく許可のデフォルトの動作ではありませんが、AS_Request、TGS_TicketReq、TGS_RenewReq、および TGS_ValidateReq イベントで使用することができます。いくつかの例を以下に示します。

- 不正なサード・パーティーのプロトコル・パスワードを指定し、ユーザーが **dce_login** と非 DCE プリンシパルを使ってログインする場合。このイベント (AS_Request 0x101) の監査レコードには、失敗の結果、**rpc_c_authz_name** の許

可状況、および nil クライアント UUID が入ります。また、イベント固有の項目だけが、ユーザーがプリンシパルとして **dce_login** に指定したストリングになります。

- 不正なサード・パーティーのプロトコル・パスワードを指定し、ユーザーが **dce_login** と DCE プリンシパルを使ってログインする場合。このイベント (AS_Request 0x101) の監査レコードには、否定の結果、**rpc_c_authz_name** の許可状況、および **dce_login** に指定したプリンシパル名を識別するクライアント UUID が入ります。
- Entrust ユーザー・プロファイルの不正な Entrust パスフレーズを指定し、ユーザーが **dce_login** と Entrust ユーザー・プロファイルの名前を使ってログインする場合。監査レコードには、否定の結果、**rpc_c_authz_name** の許可状況が入ります。 Entrust プロファイル名が有効な DCE プリンシパル名と一致しない場合、クライアント UUID はすべてゼロになり、**dce_login** に指定したプロファイル名が、監査レコードにある唯一のイベント固有項目になります。
- Entrust ユーザー・プロファイルの Entrust パスフレーズを指定し、ユーザーが **dce_login** と Entrust ユーザー・プロファイルの名前を使ってログインする場合。このイベント (AS_Request 0x101) の監査レコードには、成功の結果と **rpc_c_authz_name** の許可状況が入ります。 Entrust プロファイル名が有効な DCE プリンシパル名と一致しない場合、クライアント UUID はすべてゼロになり、**dce_login** に指定したプロファイル名が、監査レコードにある唯一のイベント固有項目になります。

監査レコードに保管される名前は、ログイン試行時にユーザーが指定した名前です。名前が偶然有効な DCE プリンシパル名であれば、そのプリンシパルの UUID が保存されます。名前が有効なものでない場合、その名前はイベント固有の情報として表示されます。 PKC ログインを使用すると、Entrust ユーザー・プロファイルが任意の DCE プリンシパル名と同じであれば、そのプリンシパルの UUID はクライアント UUID として監査レコードに保存されます。イベント固有情報は存在しません。

DCE 監査環境変数の設定

DCE 監査サービスのオペレーションと関連している環境変数は 3 つあります。アプリケーション・サーバー (つまり、DCE 監査クライアント) を実行する前に、DCE 監査環境変数を設定する必要があります。環境変数は次のとおりです。

- **DCEAUDITOFF**- アプリケーションの開始時にこの変数を宣言すると、監査はオフになります。デフォルトではこの変数は宣言されません。
- **DCEAUDITFILTERON**- アプリケーション開始時にこの変数を宣言すると、フィルターが使用可能になります。デフォルトではこの変数は宣言されません。つまり、フィルター処理は行われず、すべての監査事象が記録されます。
- **DCEAUDITTRAILSIZE**- 監査証跡の最大サイズを設定します。
- **DCEAUDITWRAP**- 最大サイズに達すると、証跡は折り返します。

監査デーモンの始動

DCE 監査サービスは分散アプリケーションではありません。クライアント・アプリケーションが監査サービスを使用する場合でも、監査デーモン (**auditd**) をすべての DCE ホストで実行する必要はありません。監査ログが中央証跡ファイルに入る場合、あるいはフィルターがホストにインストールされる場合にのみ、監査デーモンをホストで実行する必要があります。理由は、監査デーモンが中央証跡ファイルへのアクセスを制御し、そして監査フィルターも管理するためです。ただし、DTS デーモンとセキュリティー・サーバー・デーモンが監査クライアントであるため、これらのサーバーが実行するセル内のすべてのホストで監査デーモンを実行することを検討することもできます。

監査デーモンを始動できるためには、**root** である必要があります。

監査デーモンを始動するためには、次のコマンドを使用します。

```
start.dce audit
```

このコマンドは、デーモンの動作に影響を及ぼすフラグを使用します。これらのフラグに関する詳細については、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。

監査デーモンへのアクセスの制御

無許可のアプリケーション・サーバー (監査クライアント) が監査デーモンを使用することを防止するためには、監査デーモンへのアクセスを制御する必要があります。無許可サーバーがその管理レコードをログできるとすると、監査ストレージ・スペースは使い尽くされてしまいます。

dcecp を使用して、監査デーモン・オブジェクト ***/.:/hosts/****hostname****/audit-server*** の ACL を編集することによって、監査デーモンへのアクセスを制御します。

DCE 監査サービスでサポートされる DCE 許可

DCE 監査サービスは、監査デーモンの ACL の定義に使用できる、次の DCE 許可をサポートします。

r 読み取り許可。プリンシパルがフィルターを読み取ることを許可します。

w 書き込み許可。プリンシパルがフィルターを変更することを許可します。

c 制御許可。プリンシパルが監査デーモンを制御することを許可します。これには、ロギング・サービスを使用可能または使用不能にすることと、監査デーモンの ACL を変更することが含まれます。

l ログ許可。プリンシパルが監査証跡ファイルに監査レコードを書き込むことを許可します。

監査デーモンの初期 ACL

ホストの監査デーモンの初期 ACL には次のエントリーが含まれます。

```
{unauthenticated -r--}
{user hosts/nodoz/self crwl}
{group subsys/dce/audit-admin crwl}
{any_other -r--}
```

最初のエントリーは、無許可ユーザーに対してフィルターへの**読み取り**アクセスのみを許可します。 2 番目のエントリーは、ホスト・プリンシパル (**hosts/**<*hostname*>**/self**) が、フィルターの照会と変更、監査デーモンの制御、および監査証跡ファイルへの書き込みを許可します。 3 番目のエントリーは、グループ **subsys/dce/audit-admin** のメンバーが、ホスト・プリンシパルと同じアクセス権を持つことを許可します。最後のエントリーは、その他のすべてのプリンシパルに対し、フィルターへの**読み取り**アクセスのみを許可します。 **dcecp** を使用して、ユーザーのセキュリティー要件に適合するように、ACL を変更することができます。

監査クライアントおよび管理者への許可の付与

dcecp を使用して、監査証跡ファイルへの**ログ**認可を監査クライアントに付与する、監査デーモンの ACL にエントリーを追加することができます。ホスト上のサーバー (監査レコードの生成が許可されている) で構成される、 DCE セキュリティー・グループを作成することができます。たとえば、次のとおりです。

group/hosts/<*hostname*>**/audit-clients**

このグループに、監査デーモンへの**ログ**許可を与えます。たとえば、次のとおりです。

```
dcecp> acl modify /.:/hosts/machine1/audit-server ¥
> -add {group hosts/machine1/audit-clients l}
dcecp>
```

すべてのクライアントをこのグループのメンバーにして、そのグループの監査デーモンへの許可を継承させることができます。

監査デーモンへの読み取り、照会、および制御の各認可を指定した管理者に付与するためには、 ACL エントリーも追加する必要があります。たとえば、管理者のグループ **hosts/machine1/audit-admin** の場合、次のようにします。

```
dcecp> acl modify /.:/hosts/machine1/audit-server ¥
> -add {group hosts/machine1/audit-admin rwc}
dcecp>
```

イベント・クラスの定義

個々の監査イベントをグループ化して、イベント・クラスを形成することができます。イベント・クラスは、何組かのイベントを論理的にグループ化し、 1 つの値を使用して選択できるようにする、効率のよいメカニズムを提供します。

DCE 監査イベント・クラスは構成可能です。既存のイベント・クラスのイベントの追加や削除、あるいは新しいイベント・クラスの定義を行うことができます。

ローカル・イベント・クラスを定義できることは、複数の DCE アプリケーションにおける監査サービスの管理を単純化するのに役立ちます。管理者は、自身のセキュリティー要件と証跡記憶リソースの制約を反映した、独自の監査イベント・クラスを設計することができます。

セキュリティー違反を追跡するために、一時イベント・クラスを作成することもできます。

イベント・クラスを定義するステップ

イベント・クラスを定義するためには、次のステップに従います。

1. セル管理者からイベント・クラスのイベント・クラス番号を入手する。イベント・クラス番号の範囲は、 OSF によって事前にオーガニゼーションに割り振られている必要があります。割り振られていない場合には、OSF に連絡してください。
2. イベント・クラス・ファイルを、 *dcelocal*/**etc/audit/ec** ディレクトリーに作成する。ファイルを次のように編集します。
   a. 次のフォーマットの行を追加して、 イベント・クラス番号 (ECN) を宣言します。

      ```
      ECN=_event_class_number
      ```

      ECN はホスト上のすべてのイベント・クラスに固有でなければなりません。
   b. オプションで、ファイルにサーバー・イベント・プレフィックス (SEP) 行を追加することができます。 SEP 行には、各サーバーのイベント番号の接頭部が含まれています。イベント番号の接頭部は、各サーバーの最下位のイベント番号です。 SEP 行のフォーマットは次のとおりです。

      ```
      SEP=_event_number1 event_number2 event_number3 ...
      ```

      SEP 行は、ファイルの任意の場所に置くことができます。 SEP 行を使用すると、無関係のイベント・クラス・ファイルをスキップするので、監査クライアントの走査が速くなります。
   c. アプリケーションから、イベント・クラスに組み込みたいコード・ポイントのイベント番号を取得します。
   d. イベント・クラスに組み込みたいイベントに対応するイベント番号を、 1 行に 1 つずつ追加します。

イベント・クラス・ファイルでは、空の行は無視され、コメントはコメント・テキストの前の # (番号記号) によって指定されます。

イベント・クラス・ファイルの例

次に、**ec_local_cell_critical_events** という名前のイベント・クラス・ファイルの例を示します。

```
ECN = 0xC0000005
# Server Event Number Prefixes
# 0x000001 Security Service Events
# 0x000002 Time Service Events
# 0x000003 Audit Service Events
SEP = 0x00000100 0x00000200 0x00000300
# Security Service Critical Events
# evt_osf_dce_rs_properties_set_info (sets registry properties)
0x0000011f
# evt_osf_dce_rs_policy_set_info (sets registry policy)
0x00000121
# evt_osf_dce_rs_rep_admin_stop (stops the registry service)
0x00000127
# evet_osf_dce_rs_rep_admin_mkey (changes master key)
0x00000129
# Time Service Critical Events
# evt_osf_dce_dts_create (creates a server or a clerk)
0x00000201
# evt_osf_dce_dts_delete (deletes a server or a clerk)
```

```
0x00000202
# evt_osf_dce_dts_enable (enables the time service)
0x00000203
# evt_osf_dce_dts_disable (disables the time service)
0x00000204
# Audit Service Critical Events
# evt_osf_dce_aud_enable (enables audit-record logging service)
0x00000301
# evt_osf_dce_aud_disable (disables audit-record logging service)
0x00000302
# evt_osf_dce_aud_stop (terminates the execution of the audit daemon)
0x00000303
```

フィルターの作成と保守

監査デーモンを開始してイベント・クラス・ファイルを作成してから、 **dcecp** を実行し、監査デーモンで保守されるフィルターの作成、変更、または表示を行うことができます。フィルターの作成、変更、および削除を行うために、 **audfilter create**、**audfilter modify**、および **audfilter delete** の各コマンドを使用します。既存のフィルターを表示する場合には、 **audfilter catalog** および **audfilter show** コマンドを使用します。

フィルターの作成

次に、フィルターを作成するための **audfilter create** コマンドの例を示します。

```
dcecp> audfilter create  {group trust} ¥
> -attribute {ec_local_bank_audit denial log}
dcecp>
```

このコマンドの例は、ローカル・セルの **trust** という名前の DCE グループに対して、フィルター・タイプ **group** を作成するように指定します。

-attribute オプションが必要です。このオプションに対する引き数は、フィルター・ガイド またはフィルター・ガイドのリストです。それぞれのフィルター・ガイドは、3 つのエレメント、すなわち、*イベント・クラス名* またはイベント・クラス名のリスト、*監査条件* または監査条件のリスト、および*監査アクション* または監査アクションのリストで構成されます。

イベント・クラス名は、フィルターを作成するイベント・クラス・ファイルの名前と対応します。

監査条件とは、イベントを監査するために必要な条件です。有効な条件は、**success**、**denial**、**failure**、**pending**、および **all** です。

監査アクションは、生成されるイベントが指定の監査条件と一致した場合にとるアクションです。有効なアクションは、**none**、 **log**、**alarm**、**ems**、および **all** です。

フィルターの変更

1 つまたは複数のフィルターのガイドを追加または削除することによって、既存の監査フィルターを変更することができます。次に、既存のフィルターを変更するための **dcecp** コマンドの例を示します。

```
dcecp> audfilter modify world -add {Monetary_Transfers denial log}
dcecp>
```

このコマンドの例では、既存のフィルター・タイプ **world** に、イベント・クラス **Monetary_Transfers**、監査条件 **denial**、および監査アクション **log** を追加します。フィルター・タイプ **world** は、キーを持たないことに注意してください。

DCE 制御プログラムはコンマを使用しません。標準の **dcecp** リスト形式 (単一の引き数の場合は **{x y}**、または複数の引き数の場合は **{{x y} {a b}}**) では、複数のガイドと複数のフィルターが指定されます。

audfilter modify コマンドを実行するためには、監査デーモンの ACL に対する書き込み (**w**) 許可が必要です。

フィルターの削除

audfilter delete コマンドを使用して、 DCE クライアントの 1 つまたは複数の監査フィルターを削除することができます。次に、 **audfilter delete** コマンドの例を示します。

```
dcecp> audfilter delete  {foreign_principal/.../foreign_cell_name/jedwards}
dcecp>
```

このコマンドの例では、外部セル **/.../foreign_cell_name** の DCE プリンシパル **jedwards** の監査フィルターが削除されます。

audfilter delete コマンドに、削除するフィルターを複数指定することができます。上記のフィルターの変更の例の場合と同じように、複数のフィルターを削除するときは、標準の **dcecp** 構文を使用する必要があります。

このコマンドを実行するためには、監査デーモンの ACL に対する書き込み (**w**) 許可が必要です。

デフォルト・フィルター

ホストの構成時に (**dce_config** を使用)、以下の **audfilter create** コマンド (**dcecp** を使用) が実行されて、セキュリティー・デーモン、 DTS デーモン、および監査デーモンの各フィルターが作成されます。

```
audfilter create world -at {dce_sec_modify success log}
audfilter create world -at {dce_sec_modify {failure denial} all}
audfilter create world -at {dce_sec_server success log}
audfilter create world -at {dce_sec_server {failure denial} all}
audfilter create world -at {dce_sec_authent {failure denial} all}
audfilter create world -at {dce_sec_query denial all}
audfilter create world -at {dce_dts_mgt_modify success log}
audfilter create world -at {dce_dts_mgt_modify {failure denial} all}
audfilter create world -at {dce_dts_mgt_query {failure denial} all}
audfilter create world -at {dce_audit_admin_modify success log}
audfilter create world -at {dce_audit_admin_modify {failure denial} all}
audfilter create world -at {dce_audit_filter_modify success log}
audfilter create world -at {dce_audit_filter_modify {failure denial}
all}
audfilter create world -at {dce_audit_admin_query {failure denial} all}
audfilter create world -at {dce_audit_filter_query {failure denial}
all}
```

監査フィルターの使用可能化

監査フィルターを使用可能にしたい場合には、まず環境変数 **DCEAUDITFILTERON** を設定する必要があります。サーバー (つまり、監査クライアント) を始動する前に、この変数を設定する必要があります。

更新バインディング・ファイルの削除

サーバー (監査クライアント) が、フィルターを使用可能にして (すなわち、**DCEAUDITFILTERON** が設定されている) 作動している場合、 **libaudit** (サーバーにリンクされている) は、サーバーのバインド情報を入手して、それを以下に格納します。

/opt/dcelocal/var/audit/client/*pid-of-server***/update_binding_file**

ここで、*pid-of-server* はサーバーのプロセス ID です。

サーバーが異常終了した場合は、このファイルを手作業で削除する必要があります。このファイルを削除しないと、次回に **DCEAUDITFILTERON** を指定してサーバーを再始動すると、エラー・メッセージが出されます。このメッセージは、監査デーモンが、フィルター更新を監査クライアントに通知できないことを示します。

```
unable to inform process
/opt/dcelocal/var/audit/client/pid-of-server/update_binding_file
about esl update.
```

また、古い更新バインディング・ファイルがないかどうか検査することもできます。このためには、稼働しているサーバーを検査し (たとえば、**ps -e** を使用して)、そのプロセス ID と更新バインディング・ファイルのパス名を比較します。これらのファイルのパス名には、 **pid-of-server** のコンポーネントが含まれているため、存在しないサーバーに対応するファイルを判別することができます。

バインディング情報とその情報を含むディレクトリー (*pid-of-server*) の両方とも削除する必要があります。

監査証跡のバッファリング

オペレーティング・システムは、監査証跡データがディスクに書き込まれる前に、書き込み中にそれをバッファリングします。このため、監査証跡ファイルの拡大は、データがディスクにフラッシュされるまで、明らかになりません。

監査ロギング・サービスの使用可能化と使用不能化

監査デーモンの監査レコード・ロギング・サービスを使用可能または使用不能にするには、 **dcecp** を使用します。 **aud enable** コマンドはロギング・サービスを使用可能にし、 **aud disable** コマンドはそれを使用不能にします。

監査証跡ファイルが大きくなりすぎたらロギング・サービスを使用不能にし、監査証跡をバックアップして巻き戻したら (**aud rewind** コマンドを使用)、再び使用可能にすることができます。

使用可能化コマンドまたは使用不能化コマンドを使用すると、中央監査証跡ファイルへの監査レコードのロギングが使用可能または使用不能にされます。セキュリテ

ィー・サーバーおよびタイム・サーバーなどのアプリケーションは、独自の監査証跡ファイルを使用するため、使用可能化コマンドや使用不能化コマンドの使用には影響されません。

aud stop コマンドは監査デーモンを停止します。

監査デーモン属性の変更と照会

DCE 監査デーモンは、監査証跡ファイルに関連した、次の 2 つの属性を持っています。

- **stostrategy** - 監査証跡ファイルのサイズがその限界に達したときの、記憶ストラテジーを指定します。次の記憶ストラテジーのいずれかを指定することができる。

  save 指定された証跡サイズ限界に達した場合、監査デーモンは中央証跡ファイルを新しいファイルに保存する (そのファイルの名前を元の名前に変更し、タイム・スタンプをその名前の後に付加する)。次に、監査デーモンは元の証跡ファイルの内容を削除して、このファイルの先頭から監査を継続する。これが **stostrategy** のデフォルト値です。

  wrap 中央証跡ファイルが最大サイズ (**dce_aud_set_trail_size( )** または環境変数 **DCEAUDITTRAILSIZE** により設定) に達すると、そのファイルの先頭で監査デーモンが監査レコードの上書きを始めますが、ファイルを消してしまうことはありません。新しいレコードが証跡ファイルに書き込まれると、そのファイルにある最も古い監査レコードが上書きされます。

- **state** - 監査デーモンが、監査クライアントからの監査レコード・ロギング要求にサービスを提供中かどうかを指示します。この属性に指定可能な値は、 **enabled** (デフォルト値) または **disabled** です。

次のように、**dcecp** を使用して、これらの設定値を見ることができます。

```
dcecp> aud show
{state enabled}
{stostrategy save}
dcecp>
```

これらの属性を変更するためには、**aud modify** コマンドを使用します。

監査証跡の制御と表示

監査デーモンは、監査クライアントから送信された監査レコードを監査証跡ファイルに記録します。引き数を何も指定せずに監査デーモンを始動した場合に使用されるデフォルトの監査証跡ファイルは、 *dcelocal***/var/audit/adm/central_trail** です。デーモンの開始時に、 **auditd** コマンドの **-t** オプションを使用して、監査証跡の出力先を別のファイルにすることもできます。 **-t** オプションの *trail* 引き数は、ログを書き込むファイルのパス名を指定します。

監査証跡ファイルの表示

監査証跡ファイルの内容を調べるためには、 **dcecp audtrail show** コマンドを使用します。中央監査証跡ファイルまたはローカル監査証跡ファイルのいずれかの内容を表示することができます。

たとえば、次のコマンドを使用して、監査証跡ファイル **central_trail** の内容を表示することができます。

```
dcecp> audtrail show /opt/dcelocal/var/audit/adm/central_trail
--- Event Record number 31 --
o Event Information:
   - Event Number:                0x113 /* 275 */
   - Event Name:                  LOGIN_GetInfo
   - Event Outcome:               success
o Server:                         /.:/hosts/zinfandel.austin.ibm.com
o Client:                         /.../budokan_3/hosts/zinfandel.austin.ibm.com/self
o Number of groups:               0
o Authorization Status:           Authorized with a pac
o Date and Time recorded:         1999-01-19-11:48:40.538-06:00I----
o 1 Event(s) specific:
  - item number 1                 char string hosts/zinfandel.austin.ibm.com/self
--- End of Event record number 31 --
--- Event Record number 34 --
o Event Information:
   - Event Number:                0x113 /* 275 */
   - Event Name:                  LOGIN_GetInfo
   - Event Outcome:               success
o Server:                         /.:/hosts/zinfandel.austin.ibm.com
o Client:                         /.../budokan_3/hosts/zinfandel.austin.ibm.com/self
o Number of groups:               0
o Authorization Status:           Authorized with a pac
o Date and Time recorded:         1999-01-19-11:48:40.873-06:00I----
o 1 Event(s) specific:
  - item number 1                 char string hosts/zinfandel.austin.ibm.com/self
--- End of Event record number 34 --
```

監査証跡データを画面に表示するのではなく、ファイルに書き込みたい場合は、 **-to** オプションを **audtrail show** コマンド行に組み込みます。このオプションは、指定されたファイル名に監査証跡ファイルの内容を印刷します。大きな証跡ファイルの場合は、このオプションを使用されることを強くお勧めします。

監査証跡サイズの制御

デフォルトにより、監査証跡ファイルのサイズは 2 MB に制限されます。監査サービスは、証跡ファイルのサイズがこの値を超えたことを検出すると、そのファイルをクローズし、タイム・スタンプ情報を使用してファイルの新しい固有名を作成し、そして、元の名前で新しい証跡ファイルをオープンします。そして、このファイルへの新しい監査ログの書き込みを続行します。このファイルが大きくなりすぎると、このプロセスが繰り返されます。

監査証跡ファイルのサイズを変更したい場合、監査サービスを使用するアプリケーションを始動する前に、環境変数 **DCEAUDITTRAILSIZE** を必要なサイズに設定する必要があります。この環境変数を設定すると、デフォルトの 2 MB サイズの限度が指定変更されます。 **dce_aud_set_trail_size_limit()** を呼び出すことにより、環境変数の値をサーバーでオーバーライドすることができます。

たとえば、5 MB の証跡ファイル・サイズを使用したい場合、**DCEAUDITTRAILSIZE** を次のように指定します。

```
DCEAUDITTRAILSIZE 5000000
```

また、限界 (デフォルト値の 2 MB または **DCEAUDITTRAILSIZE** で設定した限界) に達した場合に、監査デーモンが中央監査証跡ファイルを繰り返し使用するように指定することもできます。このためには、 **-wrap** オプションを指定して監査デーモンを始動する必要があります。

```
auditd -wrap
```

また、旧監査レコードの値が少ないか、ない場合で、しかも比較的最近のレコードのみを保持したい場合に、このオプションを使用することもできます。

証跡サイズの限界は、 **auditd** コマンドの **-s** オプションを使用しても設定することができます。この方法で設定した限界は、デフォルトの 2 MB の限界を指定変更します。

何らかの理由で、監査証跡が限界に達する前に、監査証跡のスナップショットをとりたい場合には、ログを使用不可にして、ファイルをコピーするために、 **dcecp aud disable** コマンドを使用することができます。その後、**dcecp aud rewind** コマンドを使用して、中央監査証跡ファイルを巻き戻します。 (必要な場合には、この時点でこの監査ファイルをバックアップできることに注意してください。ただし、バックアップする場合には、監査サービスに自動的に新しい証跡ファイルを作成させて、それらをバックアップさせるのが最良の方法です。) 次に、 **aud enable** コマンドを使用して、監査デーモンのログ・サービスを再び使用可能にします。

監査証跡ファイル記憶オプションの変更

記憶ストラテジー・オプションは、監査デーモンの実行中に変更することができます。これは、中央監査証跡ファイルに対してのみ行うことができます。

次の例は、**aud modify** コマンドを使用して、監査証跡がファイルの限界に達したときに、繰り返し使用させる方法を示しています。

```
dcecp> aud modify -stostrategy wrap
dcecp>
```

このコマンドの例は、監査デーモンの記憶ストラテジー属性の値を **wrap** に変更します。

第46章 Kerberos の DCE および Secure リモート・ユーティリティーとの相互運用性

DCE セキュリティー・サービスの認証部分は、基本的に、 Kerberos ネットワーク認証システムのバージョン 5 (インターネット・エンジニアリング・タスク・フォース (IETF) RFC 1510 で記述されている) に基づいています。これによって、DCE セキュリティー・サーバーはほとんど、 Kerberos V5 のクライアントに対する Kerberos Key Distribution Center (KDC) として作動できるようになりました。 DCE の以前のリリース (バージョン 1.2.1 およびそれ以前) では、この相互運用性は、正式にサポートされておらず、文書化もされていませんでした。

この章では、DCE と Kerberos V5 の相互運用性機能と構成について説明します。以下の主要機能は、DCE の現行リリースで提供されています。

- KDC 相互運用性: DCE セキュリティー・サービスが、 Kerberos V5 クライアントのための Kerberos KDC として使用できます。
- 認証キャッシュと keytab ファイルの互換性: DCE と Kerberos V5 のアプリケーションは、データを失うことなしに、認証キャッシュと keytab ファイルを共用できます。

KDC の相互運用性

Kerberos V5 実装の必須の相互運用性要件が IETF RFC 1510 で定義されています。 DCE セキュリティー・サーバーは Kerberos V5 クライアントの DCE KDC として使用できますが、以下の制限は以前と同様に適用されます。

- DCE KDC でサポートされない次のようなアルゴリズムがある。
  - DES-CBC-MD5 暗号化
  - CRC-32 チェックサム
  - DES-MAC チェックサム
  - DES-MAC-K チェックサム
  - DES-MD5 チェックサム
- レルム間コミュニケーション (DCE セルと Kerberos V5 レルムとの間の推移的信頼を含む) は、サポートされない。
- Kerberos V5 は DCE サード・パーティーの事前認証をサポートしない。したがって、プリンシパルに付加された *pre_auth_req* ERA が、2 (**PA-ENC-THIRD-PARTY**) という値を持っている場合には、プリンシパルは DCE KDC から TGT をリトリーブできません。

証明書キャッシュと keytab ファイルの互換性

DCE クライアントと Kerberos V5 クライアントが、同一ホスト上に共存するためには、それらが証明書とキーを共用できなければならない場合があります。 DCE バージョン 1.2.2 は、この共存を可能にするために、互換性のある証明書ファイルと keytab ファイルのフォーマットを提供します。

Kerberos V5 の第 5 ベータ版には、証明書キャッシュ・ファイルについては 3 つ、 keytab ファイルについては 2 つのフォーマット・バージョンがあります。

ファイル・フォーマットすべてが相互に互換性があるとはかぎりません。たとえば、証明書キャッシュ・ファイルをバージョン 1 のフォーマットで書く、旧 Kerberos V5 クライアントは、バージョン 2 または 3 のフォーマットの証明書キャッシュを読むことはできません。 表32 と 表33 で、証明書キャッシュと keytab ファイルの互換性について説明します。

表 32. 証明書キャッシュ・ファイル

リリース	Kerberos V5 (バージョン 1)	Kerberos V5 (バージョン 2)	Kerberos V5 (バージョン 3)
DCE 1.0 または Kerberos V5 ベータ 1	はい	いいえ	いいえ
Kerberos V5 ベータ 2-3	いいえ	はい	いいえ
DCE 1.1-1.2.1 または Kerberos V5 ベータ 4	はい	はい	いいえ
DCE 1.2.2 または Kerberos V5 ベータ 5-7	はい	はい	はい

Kerberos V5 のベータ 2 版およびベータ 3 版の不良部分により、バージョン 1 のフォーマットで作成したファイルとの互換性が妨げられていましたが、これはベータ 4 で修正されました。

表 33. keytab ファイル

リリース	Kerberos V5 (バージョン 1)	Kerberos V5 (バージョン 2)
DCE 1.0 または Kerberos V5 ベータ 1	はい	いいえ
Kerberos V5 ベータ 2-3	いいえ	はい
DCE 1.1-1.2.2 または Kerberos V5 ベータ 4-7	はい	はい

DCE クライアントが使用する証明書キャッシュ・ファイルと keytab ファイルは、**dcecp hostvar set** コマンドで設定できます。両ファイルに対するデフォルト・フォーマットは、バージョン 1 です。ホスト・ベースでこれを変更することにより、Kerberos V5 アプリケーションが DCE クライアント上で稼働し、 keytab ファイルと証明書ファイルを共用できるようになります。

たとえば、 Kerberos V5 の証明書キャッシュ・ファイルのフォーマット・バージョン番号を 3 に設定するために、次の **dcecp** コマンドを使用します。

```
dcecp> hostvar set -krbccachevno 3
dcecp>
```

Kerberos V5 の keytab ファイルのフォーマット・バージョン番号を 2 に設定するためには、次のようにします。

```
dcecp> hostvar set -krbktvno 2
dcecp>
```

現在の設定を表示するためには、次のようにします。

```
dcecp> hostvar show -krbccachevno -krbktvno
dcecp>
```

詳細については、*IBM DCE for AIX and Solaris バージョン 3.2: 管理コマンド・リファレンス* を参照してください。変更内容を有効にするためには、ホスト上の DCE デーモンを停止し、再始動します。

構成コマンド (kerberos.dce)

kerberos.dce コマンドは、既存の DCE クライアントと DCE サーバーが Kerberos を使用できるようにするためのものです。

注: DCE 3.1 以降で出荷されている **config.dce** コマンドを使って構成される、スリム・クライアント以外のマシンはすべて、このコマンドを実行する必要は**ありません**。これらのステップは、自動的に実行されます。

このコマンドは、すべてのスリム・クライアントで実行し、 Kerberos 機能を使用可能にする必要があります。このコマンドは、**/etc/krb5.conf** ファイル、 **/.:/host/** および **/.:/ftp/** アカウント、およびこれらのアカウントのキー・テーブル項目を作成します。

1. 存在しているすべてのセルで、 **-type admin** オプションを指定してコマンドを実行し、 **/.:/hosts/not_reg_mach_addrs** を作成します。この項目を作成する必要があるのは、一度だけです。
2. この機能が使われるセルにあるそれぞれのマシンで、 **-type local** オプションを指定してコマンドを実行します。こうすると、**/.:/hosts/not_reg_mach_addrs** 項目に、それぞれの IP ホスト・アドレス同士の結び付けが登録されます。
3. 定期的に、**-type admin** オプションを指定したコマンドを実行して、**/.:/hosts/not_reg_mach_addrs** 項目の下に登録したそれぞれの結び付けごとにホストと FTP アカウントを作成します。
4. この機能を使用するセルにあるそれぞれのマシンで、管理者がステップ 3 を完了した後再び、 **-type local** オプションを指定したコマンドを実行します。こうすると、このマシンの各アカウントごとにキー・テーブル項目が作成されます。

管理者が選択する場合、このコマンドは **-type admin** および **-ip_name** オプションを指定して実行することができます。これは、前述のステップ 2 とステップ 3 で実行したステップを行うということです。

DCE/Kerberos の相互運用性拡張機能

以下の DCE/Kerberos 相互運用性拡張機能が、 DCE 3.1 for AIX および Solaris に搭載されています。

- KDC 相互運用性: DCE セキュリティー・サーバーが、 Kerberos V5 クライアントのための Kerberos KDC として使用できます。
- 認証キャッシュと keytab ファイルの互換性: DCE と Kerberos V5 のアプリケーションは、データを失うことなしに、認証キャッシュと keytab ファイルを共用できます。

追加の拡張機能は、セキュア AIX V4.3 リモート・サービス・コマンド (**rlogin**、**rsh**、**rcp**、**telnet**、および **ftp**) をサポートすることにより、 DCE/Kerberos 相互運用性拡張機能を拡張します。また、ユーザー・アプリケーションで使用することもできます。これらの拡張機能については *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド − コア・コンポーネント* では説明されていませんが、以下で説明

されています。リモート・サービス・コマンドの詳細については、 AIX Version 4.3 System User's Guide の *Communications and Networks* セクションを参照してください。

- GSSAPI 相互運用性: DCE アプリケーションは、GSSAPI コンポーネントを使用して、 Kerberos V6 アプリケーションの GSSAPI コンポーネント・ワイヤーを通じて相互に運用することができます。
- **dce_login** および **dceunixd** を指定した Kerberos フラグのサポート : DCE **dce_login** および **dceunixd** コマンドは、 Kerberos の更新可能、転送可能、または代理可能なフラグを指定して、認証を取得することができます。 **kinit** は日付を実際より遅らせて設定するのに使用できます。
- **k5dcelogin** コマンドおよび API: アプリケーションは、 **k5dcelogin** コマンドまたは API を使用して、 Kerberos V5 認証を DCE 認証に昇格させることができます。こうすることにより、分散ファイル・システム (DFS) ファイルなどの DCE オブジェクトにアクセスするために、この認証を使用することができます。
- 構成および管理サポート : DCE では、 Kerberos V5 **krb5.conf** ファイルの構成と更新に使用できるツールが用意されているだけでなく、セキュア AIX V4.3 リモート・サービスのコマンドによって使用されるアカウントも準備されています。

DCE GSSAPI コンポーネントは、次のような DCE/ Kerberos 相互運用性のシナリオをサポートするように拡張されています。

- すべてのフラグおよびパラメーター付きの Kerberos **gss_init_sec_context** および DCE **gss_accept_sec_context** 間での、有線による相互運用性がサポートされます。
- すべてのフラグおよびパラメーター付きの DCE **gss_init_sec_context** および Kerberos **gss_accept_sec_context** 間での、有線による相互運用性がサポートされますが、 **GSS_C_DELEG_FLAG** ではサポートされていません。
- すべてのフラグおよびパラメーター付きの Kerberos **gss_seal** および DCE **gss_unseal** 間での、有線による相互運用性がサポートされます。
- すべてのフラグおよびパラメーター付きの DCE **gss_seal** および Kerberos **gss_unseal** 間での、有線による相互運用性がサポートされます。

このような相互運用性のシナリオをサポートするために、DCE では、新しいメカニズム・タイプ **GSS_MECH_MIT_KRB5** がサポートされるようになりました。このメカニズム・タイプには、**gss_mech_krb5** の場合と同じ OID 値と動作があります。これは、 Kerberos V5 リリース 1 で使用するデフォルトのメカニズムです。

後方互換性のために、IBM DCE V3.2 for AIX は、引き続き **GSS_C_OID_KRBV5_DES** をサポートしています。これは、以前のリリースの DCE に備えられていた Kerberos メカニズム・タイプです。ただし、**GSS_C_OID_KRBV5_DES** メカニズム・タイプでは、変更されていないリリースの Kerberos V5 との相互運用性はなくなっています。

新しい **GSS_MECH_MIT_KRB5** メカニズム・タイプは、メカニズム・タイプのパラメーターを持つ各 GSSAPI API によってサポートされています。次に、これらの API と、その API で **GSS_MECH_MIT_KRB5** メカニズム・タイプを使用する方法についての情報を示します。

- **gss_accept_sec_context** (出力 **actual_mech_type パラメーター**)

Kerberos V5 メカニズムを使用したコンテキストが起動側に受け入れられた場合、 **GSS_MECH_MIT_KRB5** を戻します。

- **gss_acquire_cred** (入力 **desired_mechs パラメーター**)

  Kerberos V5 メカニズムを使用するコンテキストを受け入れ側で開始する目的で使用される **GSS_C_INITIATE** または **GSS_C_BOTH** タイプの証明書を獲得する場合、**GSS_MECH_MIT_KRB5** と指定します。

  注: **GSS_C_ACCEPT** タイプの証明書を獲得する場合、メカニズム・タイプ・パラメーターは使用されません。このことは、すべてのメカニズム・タイプに当てはまります。

- **gss_display_status** (入力 **mech_type パラメーター**)

  Kerberos V5 R1 メカニズムを使用するコンテキストの状況を表示する場合、**GSS_MECH_MIT_KRB5** と指定します。

- **gss_indicate_mechs** (出力 **mech_set パラメーター**)

  DCE GSSAPI でサポートされるメカニズムの 1 つとして、**GSS_MECH_MIT_KRB5** を含むメカニズム・セットを戻します。

- **gss_init_sec_context** (入力 **mech_type パラメーター**)

  Kerberos V5 メカニズムを使用するコンテキストを受け入れ側で開始する場合、**GSS_MECH_MIT_KRB** と指定します。

- **gss_inquire_cred** (出力 **mech パラメーター**)

  証明書を使用して Kerberos V5 受け入れ側でコンテキストを開始できる場合、**GSS_MECH_MIT_KRB5** を含むメカニズム・セットを戻します。

- **gssdce_login_context_to_cred** (入力 **desired_mechs パラメーター**)

  Kerberos V5 GSSAPI 受け入れ側でコンテキストを開始するために使用される証明書を獲得する場合、 **GSS_MECH_MIT_KRB5** と指定します。 (複数のメカニズム・タイプを指定できます。)

付録A. CDS の有効文字およびネーミング規則

この付録では、CDS インターフェースが使用する DCE ディレクトリー・サービス名のための有効な文字セットについて説明します。また、特殊な意味を持つ文字についても述べ、大・小文字の突き合わせ、構文、およびサイズ制限に関する制約事項と規則についても説明します。これは、CDS と DNS に対する包括的な参考資料ではありませんが、各サービスに関して記憶すべきいくつかのキー・ポイントについてその概要を示しています。 DNS 名における有効文字に関する詳細については、それぞれのテクノロジーの資料を参照してください。

DCE で名前を使用すると、複数のディレクトリー・サービスが関係することがよくあります。たとえば、CDS は、DNS のいずれかと対話して、ローカル・セルの外部で名前を検出します。

注: CDS と DNS は、すべて独自の有効文字セットと構文規則を持っているため、問題を回避する最良の方法は、名前は短く単純にして、 3 つのサービスすべてに共通する最小の文字セットで構成することです。お勧めするセットは、A から Z、a から z の文字、および 0 から 9 の数字です。このサブセットを使用することで、ディレクトリー・サービスの相互運用が容易になることに加え、種類の異なるハードウェアとソフトウェアの環境にあるユーザーが、名前の作成と使用において問題に出会う可能性が低くなります。

582ページの図78 は、CDS 名における有効文字、および CDS インターフェースが使用する DNS 名における有効文字について詳細に説明しています。

- 白枠内の文字は、3 種類の名前すべてに有効です。
- 薄い陰枠内の文字は、CDS 名においてのみ有効です。
- 濃い陰枠内の文字は、CDS 名においてのみ有効です。

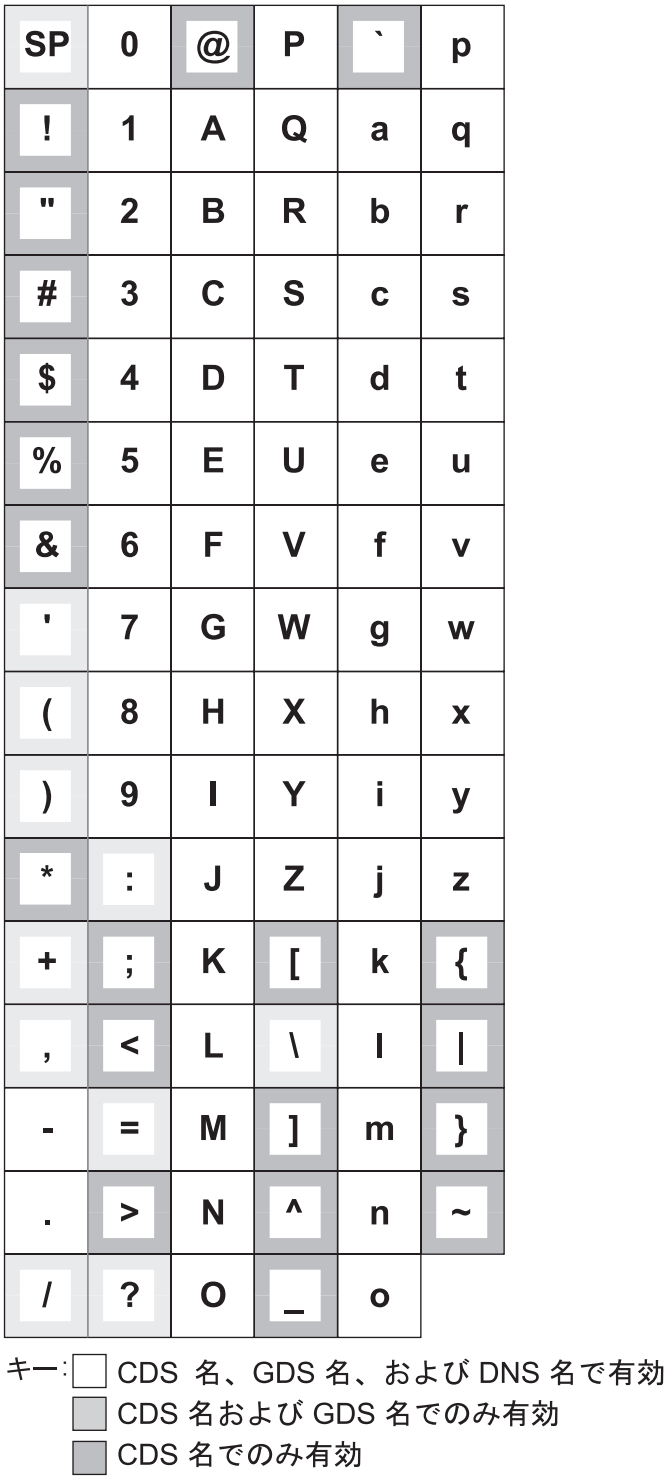


図 78. CDS 名および DNS 名の有効文字

スペースは、CDS 名において有効ですが、スペースを含む CDS 単純名は、 CDS 制御プログラムをとおして入力するときは、『』 (引用符) で囲む必要があります。その他のインターフェース固有の規則については、該当する箇所に記載してあります。

メタキャラクター

ある種の文字は、ディレクトリー・サービスに対して特殊な意味を持っています。これらの文字は、メタキャラクター と呼ばれます。 表34 は、 CDS および DNS の各メタキャラクターをリストし、説明しています。

表 34. メタキャラクターとその意味

ディレクトリー・サービス	文字	意味
CDS	/	名前のエレメントを区切ります (単純名)。
	¥	必要な場合、**¥** (円記号) の前で使用して、その文字をエスケープします。また、その後に続く文字がメタキャラクターでないことを指示します。
	,	RDN 内の複数の属性型 / 値のペア (属性値代入) を区切ります。
	¥=	属性値代入において属性型と値を区切ります。
	¥	**/** (スラッシュ)、**,** (コンマ)、または **=** (等号) の前で必要に応じて使用し、その文字をエスケープします。また、次の文字がメタキャラクターではないことを指示します。
DNS	.	名前のエレメントを区切ります。

一部のメタキャラクターは、名前の中では通常の文字として許可されていません。円記号メタキャラクターでエスケープすれば、その他のメタキャラクターは名前の中で通常の文字として使用することができます。

名前の最大サイズ

表35 は、ディレクトリー・サービス名の最大サイズをリストしています。この限界が、アーキテクチャーではなく、実装固有のものであることにご注意ください。

表 35. ディレクトリー・サービス名の最大サイズ

名前のタイプ	最大サイズ (文字)
CDS 単純名 (2 つのスラッシュの間のストリング)	254
CDS フルネーム (グローバル接頭部またはローカル接頭部、セル名、および単純名を分離するスラッシュを含む)	1023
DNS 相対名 (2 つのドットの間のストリング)	64
DNS 完全修飾名 (すべての相対名の和)	255

付録B. オブジェクト識別子ファイル

X/Open ディレクトリー・サービス (XDS) インターフェースにより、クライアント・アプリケーション・プログラマーは、 CDS で名前の作成と保守ができるようになります。また、プログラマーは、新しい CDS 属性名、CDS 属性型ラベルを作成することもできます。 DCE バージョン 1.1 ディレクトリー・サービスにおいて、 CDS 属性名は、すべて、*オブジェクト ID* (OID) と呼ばれる、対応する固有の番号を持っています。

CDS は、オブジェクト識別子と人間が読める名前との間の変換を行う手段を提供します。この変換機能により、ユーザーが DCE 制御プログラム (**dcecp**) インターフェースで、オブジェクト識別子の代わりに名前を入力することが可能になります。また、**dcecp** はコマンド出力に、オブジェクト識別子はなく、名前を表示します。CDS 属性名およびその対応する識別子は、**cds_attributes** と呼ばれるファイルに格納されます。 (すべての CDS ファイルの完全パス名については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド － 入門* と *OSF DCE Porting and Testing Guide* を参照してください。)

この付録では、**cds_attributes** および **cds_globalnames** ファイルの内容と使用法について説明し、アプリケーション開発者またはディレクトリー・サービス管理者が新しい属性のオブジェクト識別子を持つファイルを更新する方法について説明します。

オブジェクト識別子の起点

オブジェクト識別子の目的は、多種多様なアプリケーションが生成し、使用する属性型において固有性を確保することです。オブジェクト識別子は通常、最上位が国際標準化機構 (ISO) および国際電信電話諮問委員会 (CCITT) である、階層形式の割り振り機関から取得します。個々のアプリケーション開発者は通常、 ISO または CCITT に直接連絡をとって固有の番号を得る必要はありません。アプリケーション開発者は、多くの場合、割り振りを担当する社内の担当者にオブジェクト識別子を請求します。会社の責任者がその上の管理機関に連絡をとり、会社の固有の接頭部を取得します。

階層形式の割り振り機関は、オブジェクト識別子の部分を分離するドットにより指示されます。ドットにより区切られた各数字列は、左から右へ最上位機関から下位への、割り振り機関のレベルを表しています。たとえば、オブジェクト識別子 **1.3.22.1.1.2** は、次に示すレベルで構成されています。

1 ISO
3 識別されたオーガニゼーション
22 オープン・ソフトウェア・ファウンデーション (OSF)
1 分散コンピューティング環境
1 リモート・プロシージャー呼び出し
2 RPC オブジェクト UUID

cds_attributes ファイル

cds_attributes ファイルには、 CDS 属性およびオブジェクト・クラスのオブジェクト識別子が入っています。以下に示すのは、ファイルのデフォルトの内容の一部のサンプルです。

```
#     OID       LABEL                   SYNTAX
#
1.3.22.1.3.10   CDS_Members             GroupMember
1.3.22.1.3.11   CDS_GroupRevoke         Timeout
1.3.22.1.3.12   CDS_CTS                 Timestamp
1.3.22.1.3.13   CDS_UTS                 Timestamp
1.3.22.1.3.15   CDS_Class               byte
1.3.22.1.3.16   CDS_ClassVersion        Version
1.3.22.1.3.17   CDS_ObjectUUID          uuid
1.3.22.1.3.19   CDS_Replicas            ReplicaPointer
1.3.22.1.3.20   CDS_AllUpTo             Timestamp
1.3.22.1.3.21   CDS_Convergence         small
1.3.22.1.3.22   CDS_InCHName            small
1.3.22.1.3.23   CDS_ParentPointer       ParentPointer
1.3.22.1.3.24   CDS_DirectoryVersion    Version
1.3.22.1.3.25   CDS_UpgradeTo           Version
1.3.22.1.3.27   CDS_LinkTarget          FullName
1.3.22.1.3.28   CDS_LinkTimeout         Timeout
1.3.22.1.3.30   CDS_Towers              byte
1.3.22.1.3.32   CDS_CHName              FullName
1.3.22.1.3.34   CDS_CHLastAddress       byte
1.3.22.1.3.36   CDS_CHState         small
1.3.22.1.3.37 CDS_CHDirectories       CHDirectory
1.3.22.1.3.40   CDS_ReplicaState        small
1.3.22.1.3.41   CDS_ReplicaType         small
1.3.22.1.3.42   CDS_LastSkulk           Timestamp
1.3.22.1.3.43   CDS_LastUpdate    Timestamp
1.3.22.1.3.44   CDS_RingPointer         uuid
1.3.22.1.3.45   CDS_Epoch               uuid
1.3.22.1.3.46   CDS_ReplicaVersion      Version
1.3.22.1.3.48   CDS_NSCellname         char
1.3.22.1.3.52   CDS_GDAPointers         gdaPointer
1.3.22.1.3.53   CDS_CellAliases         GroupMember
1.3.22.1.3.54   CDS_ParentCellPointers  ReplicaPointer
1.3.22.1.1.1    RPC_ClassVersion        byte
1.3.22.1.1.2    RPC_ObjectUUIDs         byte
1.3.22.1.1.3    RPC_Group               byte
1.3.22.1.1.4    RPC_Profile             byte
1.3.22.1.1.5    RPC_Codesets            byte
1.3.22.1.5.1    SEC_RepUUID             byte
```

最初のカラムには OID、 第 2 カラムにはラベル (識別子がマップされる名前) がそれぞれ入っており、第 3 カラムはデータ型を示します。 CDS データ型の説明は、 **cdsclerk.h** ヘッダー・ファイルにあります。 (すべての CDS ファイルの完全パス名については、 *IBM DCE for AIX and Solaris バージョン 3.2: 管理ガイド ― 入門* と *OSF DCE Porting and Testing Guide* を参照してください。)

アプリケーション・プログラマーは、言語翻訳を目的とする場合を除いては、 **cds_attributes** ファイル内の固有の OID に関連する CDS ラベルを変更する必要はありません。しかし、プログラマーは適切な割り振り機関から新規の OID を取得し、それ自身のオブジェクト・エントリーの新規属性を作成し、そして既存のリストにそれらを追加することができます。

ファイルの変更

プログラマーがディレクトリー・サービスを使用するアプリケーションを開発する場合は、ディレクトリー・サービス管理者またはアプリケーション開発者は、新規アプリケーションが使用する新規 CDS 属性名の固有の識別子を取得して、適切なファイルを更新する必要があります。

アプリケーションが CDS 内に名前を格納する場合は、 **cds_attributes** ファイルを編集してください。 (適切なデータ型記述子のリストについては、 **cdsclerk.h** ファイルを参照してください。)

注: **cds_attributes** ファイルおよび **cds_globalnames** ファイル内の標準属性の OID 値を変更する場合、他のディレクトリー・サービスの実装システムとの相互運用の問題が生じる場合があります。

CDS エンティティーの属性の変更

すべての CDS エンティティーには属性があります。属性は、そのエンティティーに関連したデータの断片または集合です。属性は、エンティティーの動作を反映または影響を与えたり、エンティティーが最後に使用可能にされた以後に発生した特定のイベントや問題の回数を記録したり、あるいはエンティティーを他のエンティティーと固有に識別したりすることができます。

CDS 属性は、ISO OID によって識別されます。 すべての CDS 属性名は、OID および対応するデータ型にマップします。通常、クライアント・アプリケーションは、属性の名前とそのデータ型を定義します。アプリケーション・プログラマーは、 **cds_attributes** ファイル内の固有の OID に関連する既存の CDS ラベルを変更する必要はありません (言語翻訳を目的とする場合を除きます)。ただし、プログラマーは適切な機関から新規 OID を取得し、プログラマー自身のオブジェクト・エントリーの新規属性を作成し、それらを既存のリストに追加することができます。各属性の OID とデータ型は、ファイル **cds_attributes** に格納されます。アプリケーションが使用できる CDS データ型の記述は、 **cdsclerk.h** ファイルにあります。

新規属性の追加

新規属性をオブジェクト・エンティティーに追加するためには、 **-add** オプションを指定した **dcecp modify** オペレーションを使用します。

新規属性を追加するためには、セル内の各ホスト上の **cds_attributes** ファイルに、事前にその新規属性を追加しておく必要があります。また、新規属性を追加しようとする先のエンティティーに対する書き込み許可も必要です。

たとえば、次のコマンドは、単一値の属性 (**owner**) を、ディレクトリー (**/.:/admin**) に追加し、その新規属性に値 **Leland** を割り当てます。

```
dcecp> directory modify /.:/admin -add {owner Leland}
dcecp>
```

次のコマンドは、新規の複数値属性 (**vegetables**) を、オブジェクト (**/.:/admin/garden**) に追加し、その新規属性に、値 **carrots** と **lettuce** を割り当てます。

```
dcecp> object
modify  /.:/admin/garden -add {vegetables {carrots} {lettuce}}
dcecp>
```

既存の属性値の変更

既存の属性値を変更するためには、 **-change** オプションを指定した **dcecp modify** オペレーションを使用します。

属性の値を変更するためには、変更しようとする属性を持つ名前に対する書き込み許可が必要です。

たとえば、次のコマンドは、**/.:/admin** ディレクトリーの所有者属性の値を、**Leland** から **Peters** に変更します。

```
dcecp> directory modify  /.:/admin -change {owner Peters}
dcecp>
```

属性の削除

属性をオブジェクト・エンティティーから削除するためには、 **-remove** オプションを指定した **dcecp modify** オペレーションを使用します。

属性を削除するためには、削除する属性を持つ名前に対する書き込み許可が必要です。

属性を削除するためには、**-remove** オプションと **-types** オプションを指定した **modify** コマンドを使用します。たとえば、以下のコマンドは、**/.:/admin** ディレクトリーから所有者属性を削除します。

```
dcecp> directory modify /.:/admin -remove owner -types
dcecp>
```

複数値属性から単一の値を削除するためには、 **-remove** オプションを使用して、削除する値を指定します。たとえば、次のコマンドは、**/.:/admin/garden** オブジェクトから属性 vegetables の値 carrot を削除します。

```
dcecp> object modify  /.:/admin/garden -remove {vegetables
carrots}
dcecp>
```

付録C. タイム・プロバイダーとタイム・サービス

この付録では、タイム・プロバイダーの選択のときに使用する基準について述べ、時刻配布サービス、タイム・プロバイダー (ハードウェアとソフトウェア)、およびタイム・プロバイダーの DTS との対話について説明します。この付録には、世界タイムゾーン・マップも記載されています。

時刻ソース選択の基準

ネットワークのための時刻ソースを選択する前に、以下の質問をしてください。

- 提供される時刻はどれだけ正確か ?

  正確さは、時刻ソース自体と、伝送メディアの影響を受けます。誤差が分かっていれば、補正することができます。

- 時刻ソースの信頼性は ?

  時刻ソースは使用可能でなければなりません。時刻ソースが使用可能でない場合、タイム・プロバイダーに接続されたサーバーは、他のサーバーからの時刻を使用し、時刻ソースが再び使用可能になると時刻の差を補正します。

- 提供範囲はどの程度か ?

  時刻ソースは、タイム・プロバイダー・サーバーが置かれている地域で使用できなければなりません。

- 既知の誤差のレベルは ?

  誤差が分かっていれば、DTS は誤差を補正できます。大部分のソースには、既知の誤差レベルがあります。

- 費用は ?
- 時刻ソースは操作環境に適合するか ?

  使用可能な電源装置と物理的条件が、時刻ソースと互換性がなければなりません。仕様については、メーカーに問い合わせてください。

表36 は、各種の時刻ソースの選択基準について要約しています。

表 36. タイム・プロバイダーの選択基準

型	提供範囲	誤差	費用
電話			
NIST	局地的	10 ミリ秒	通話あたりの 可変料金
ラジオ			
MSF	ヨーロッパ	10 ミリ秒	$1K ～ 2K
WWV	北アメリカ	100 ミリ秒	$1K ～ 2K
WWVB	北アメリカと ヨーロッパ	10 ミリ秒	$1K ～ 2K
WWVH	東部太平洋、 中部北太平洋	100 ミリ秒	$1K ～ 2K
サテライト			
GOES	全世界	1 ミリ秒。修正済み	$2K ～ $20K

表 36. タイム・プロバイダーの選択基準 (続き)

型	提供範囲	誤差	費用
GPS	全世界	< 100 ナノ秒	$15K ～ $20K

協定世界時のソース

以下のサブセクションで説明するように、UTC 時刻には、電話、ラジオ、およびサテライトなど、複数のソースがあります。

電話サービス

電話によるタイム・プロバイダー・サービスでは、タイム・プロバイダーがモデムをとおして、中央の UTC 時刻ソースにダイヤルする必要があります。モデム・スピードと回線遅延が、戻される時刻の正確さに影響する可能性があります。

電話サービスは、通常、標準機関によって提供されます。たとえば、米国では、このサービスは連邦情報・技術局 (NIST) によって提供されます。モデム・ソフトウェアの価格に加え、このサービスにはパー・コール料金がかかります。

ラジオ伝送

DTS は、ラジオ時刻ソースから時刻を取得できます。時刻と周波数の放送をモニターする市販の受信機であれば、タイム・プロバイダー・インターフェース (TPI) をとおして、時刻値を DTS サーバーに戻すことができます。 NIST は、以下の米国の時刻と周波数の局を運営しています。

- WWV

  北米と南米向けに 2.5、5.0、10.0、15.0 MHz で送信しています。
- WWVB

  主として米国向けに 60 kHz で送信しています。大気の伝搬作用が比較的小さいため、高品質の周波数情報を提供します。
- WWVH

  アラスカ、ハワイ、オーストラリア、ニュージーランド、日本、および東南アジア向けに、2.5、5.0、10.0、15.0 MHz で送信しています。

ヨーロッパでは以下の局が使用可能です。

- MSF

  英国から 60 kHz で放送しています。
- DCF77

  ドイツから 77.5 kHz で放送しています。

上記にリストした局に加え、世界の 30 以上のラジオ局が UTC 時刻を提供しています。詳細については、ユーザーの国の国内標準組織に問い合わせてください。

ネットワーク・タイム・プロトコル

インターネットのアクセスを持つノードは、DTS 用の UTC 時刻のソースとして、ネットワーク・タイム・プロトコル (NTP) を使用できます。 (NTP をタイム・プロ

バイダーとして使用する方法については、307ページの『第26章 ネットワーク・タイム・プロトコルとの相互運用』を参照してください。)

サテライト

サテライトは世界中で利用できます。遅延が分かっていて補正すれば、比較的正確な時刻を提供できます。以下の UTC のサテライト・ソースのリストを参照してください。

- GOES

  Geostationary Operational Environment Satellite

- TRANSIT

  4 つの追跡システムと 2 つの地上サテライト通信サイトから構成される、米国海軍のサテライト・システム。

- GPS

  Global Positioning System、サテライト受信局

世界のタイムゾーン・マップ

図79 は、以下を含んだ、世界タイムゾーンのマップを示しています。

- UTC 基準ゾーン
- 奇数番号と偶数番号のゾーン
- 30 分のゾーン
- ゾーン・システムを未採用の国と地域、または隣接するゾーンとの差が 30 分以外の場所

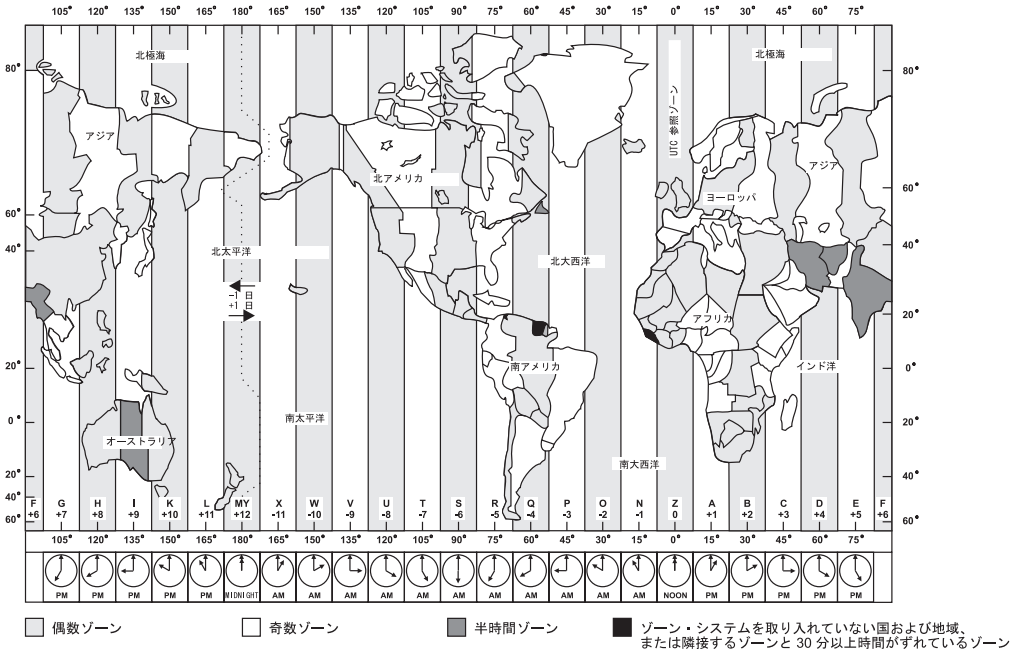


図 79. 世界のタイムゾーン・マップ

付録D. DTS 拡張 BNF

この付録では、拡張バッカス正規形式 (BNF) 表記により、分散タイム・サービス (DTS) 構文が定義されています。

DTS 時刻変換のための BNF は 4 つの部分からなっています。すなわち、 *year*、*day*、*tdf*、および *inaccuracy (誤差)* です。値が明示的に表示されていない部分には、現在日のその値として変換デフォルト値がとられます。 DTS 時刻の変換のための BNF は次のとおりです。

```
dts_time :  year_part day_part tdf_part inacc_part
|   year_part day_part tdf_part
|   year_part day_part
|   year_part day_part inacc_part
|   year_part inacc_part
|   year_part
|   day_part  tdf_part inacc_part
|   day_part  tdf_part
|   day_part inacc_part
|   day_part
|   year_part Z
|   year_part Z inacc_part
|   year_part day_part Z inacc_part
|   day_part  Z inacc_part
|   day_part  Z
;
year_part : number - number - number -
| number - number - number T
| number - number T
| number T
;
day_part  : partial : partial : partial
| partial : partial
| partial
;
tdf_part  : sign number : number
| sign number
;
sign      : -
| +
;
partial   : number
| number frac
| number frac number
| frac number
;
frac      : .
| ,
;
inacc_part : I
| I partial
| I infinity
;
infinity   :  'i''n''f'
|  - -
|  - - - - -
;
number     :  DIGIT
|  number DIGIT
;
```

付録E. 特記事項

本書において、日本では発表されていない IBM 製品 (機械およびプログラム)、プログラミングまたはサービスについて言及または説明する場合があります。しかし、このことは、弊社がこのような IBM 製品、プログラミングまたはサービスを、日本で発表する意図があることを必ずしも示すものではありません。 本書で IBM ライセンス・プログラムまたは他の IBM 製品に言及している部分があっても、このことは当該プログラムまたは製品のみが使用可能であることを意味するものではありません。 IBM 製品、プログラム、またはサービスに代えて、 IBM の有効な知的所有権またはその他の法的に保護された権利を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM によって明示的に指定されたものを除き、他社の製品と組み合わせた場合の操作の評価と検証はお客様の責任で行っていただきます。

IBM は、本書で解説されている主題について特許権 (特許出願を含む)、商標権、または著作権を所有している場合があります。本書の提供は、これらの特許権、商標権、および著作権について、本書で明示されている場合を除き、実施権、使用権等を許諾することを意味するものではありません。実施権、使用権等の許諾については、下記の宛先に、書面にてご照会ください。

〒106-0032 東京都港区六本木 3 丁目 2-31
AP 事業所
IBM World Trade Asia Corporation
Intellectual Property Law & Licensing

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

本書に対して、周期的に変更が行われ、これらの変更は、文書の次版に組み込まれます。 IBM は、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム（本プログラムを含む）との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
Department LZKS
11400 Burnet Road
Austin, TX 78758
U.S.A.

本プログラムに関する上記の情報は、適切な条件の下で、使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、 IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、 IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストはおこなっておりません。また、IBM 以外の製品に関するパフォーマンスの正確性、互換性、またはその他の要求は確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権ライセンス :

この情報には、ソース言語でのサンプル・アプリケーション・プログラムが含まれており、様々なオペレーティング・プラットフォームでのプログラミング・テクニックを示しています。サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースにかなうアプリケーション・プログラムを開発、使用、購買、または配布する目的で、これらのサンプル・プログラムを IBM への支払い義務なく、自由にコピー、変更、および配布することができます。

これらの例は、あらゆる条件を想定して徹底的にテストされているわけではありません。そのため、IBM は、これらのプログラムの信頼性、保守容易性、または機能

について保証することはできません。 IBM のアプリケーション・プログラミング・インターフェースにかなうアプリケーション・プログラムを開発、使用、購買、または配布する目的で、これらのサンプル・プログラムを IBM への支払い義務なく、自由にコピー、変更、および配布することができます。

各コピー、これらのサンプル・プログラムのすべての部分、または派生する作業すべては、以下の著作権表示に含まれる必要があります。

© (お客様の会社名) (年)。このコードの一部は、IBM Corp から派生しています。サンプル・プログラム。© Copyright IBM Corp. *1990, 2001*. All rights reserved.

お客様がソフトコピー、写真、カラーの図を表示する場合、表示されない可能性があります。

商標

次のものは、 IBM Corporation の米国およびその他の国における商標です。

- AIX
- DFS
- Encina
- IBM
- MVS

UNIX は、The Open Group がライセンスしている米国およびその他の国における登録商標です。

Microsoft、Windows、Windows NT、および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標または登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

［サ行］

[タ行]

[ナ行]

[ハ行]

［マ行］

［ヤ行］

［ラ行］

［ワ行］

［数字］

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

W

X

［特殊文字］

Printed in Japan

日本アイ・ビー・エム株式会社
〒106-8711　東京都港区六本木3-2-12