OSF® DCE Administration Guide—Introduction

Release 1.2.2

Open Software Foundation
11 Cambridge Center
Cambridge, MA 02142

Contents

Part 2. Configuring and Starting Up DCE

List of Figures

List of Tables

Preface

The *OSF DCE Administration Guide* provides concepts and procedures that enable you to manage the OSF® Distributed Computing Environment (DCE). Basic OSF DCE terms are introduced throughout the guide. A glossary for all of the DCE documentation is provided in the *Introduction to OSF DCE*. The *Introduction to OSF DCE* helps you to gain a high-level understanding of the DCE technologies and describes the documentation set that supports DCE.

Audience

This guide is written for system and network administrators who have previously administered a UNIX environment.

Applicability

This revision applies to the OSF DCE Release 1.2.2 offering and related updates. (See your software license for details.)

Purpose

The purpose of this guide is to help system and network administrators to plan, configure, and manage DCE. After reading the guide, you will understand what the system administrator needs to do to plan for DCE. Once you have built the DCE source code on your system, use this guide to assist you in installing executable files and configuring DCE. The *OSF DCE Release Notes* contain instructions for installing and building DCE source code.

Document Usage

The *OSF DCE Administration Guide* consists of two books, each of which is divided into parts, as follows:

- The *OSF DCE Administration Guide—Introduction*
  - — Part 1. Introduction to DCE Administration
  - — Part 2. Configuring and Starting Up DCE
- The *OSF DCE Administration Guide—Core Components*
  - — Part 1. The DCE Control Program
  - — Part 2. DCE Administration Tasks
  - — Part 3. DCE Host and Application Administration
  - — Part 4. DCE Cell Directory Service
  - — Part 5. DCE Distributed Time Service
  - — Part 6. DCE Security Service

Related Documents

For additional information about the Distributed Computing Environment, refer to the following documents:

- *Introduction to OSF DCE*

- *OSF DCE Command Reference*
- *OSF DCE Application Development Reference*
- *OSF DCE Application Development Guide—Introduction and Style Guide*
- *OSF DCE Application Development Guide—Core Components*
- *OSF DCE Application Development Guide—Directory Services*
- *OSF DCE DFS Administration Guide and Reference*
- *OSF DCE GDS Administration Guide and Reference*
- *OSF DCE/File-Access Administration Guide and Reference*
- *OSF DCE/File-Access User's Guide*
- *OSF DCE Problem Determination Guide*
- *OSF DCE Testing Guide*
- *OSF DCE/File-Access FVT User's Guide*
- *Application Environment Specification/Distributed Computing*
- *OSF DCE Release Notes*

For a detailed description of OSF DCE documentation, see the *Introduction to OSF DCE*.

Typographic and Keying Conventions

This guide uses the following typographic conventions:

Bold — **Bold** words or characters represent system elements that you must use literally, such as commands, options, and pathnames.

Italic — *Italic* words or characters represent variable values that you must supply. *Italic* type is also used to introduce a new DCE term.

`Constant width` — Examples and information that the system displays appear in `constant width` typeface.

[ ] — Brackets enclose optional items in format and syntax descriptions.

{ }	Braces enclose a list from which you must choose an item in format and syntax descriptions.
|	A vertical bar separates items in a list of choices.
< >	Angle brackets enclose the name of a key on the keyboard.
...	Horizontal ellipsis points indicate that you can repeat the preceding item one or more times.

This guide uses the following keying conventions:

<Ctrl-*x*> or **^*x***
The notation **<Ctrl-*x*>** or **^*x*** followed by the name of a key indicates a control character sequence. For example, **<Ctrl-C>** means that you hold down the control key while pressing **<C>**.

<Return> The notation **<Return>** refers to the key on your terminal or workstation that is labeled with the word Return or Enter, or with a left arrow.

Problem Reporting

If you have any problems with the software or documentation, please contact your software vendor's customer service department.

Pathnames of Directories and Files in DCE Documentation

For a list of the pathnames for directories and files referred to in this guide, see the *OSF DCE Administration Guide—Introduction* and *OSF DCE Testing Guide*.

Part 1

Introduction to DCE System Administration

Chapter 1

Introduction to DCE for Administrators

The *Introduction to OSF DCE* introduced you to the OSF Distributed Computing Environment (DCE), describing the major components of its services. This chapter provides an overview of DCE from the perspective of the system or network administrator.

As the *Introduction to OSF DCE* explains, DCE is a set of services that together make up a high-level coherent environment for developing and running distributed applications. These services include a set of tools that support DCE management tasks. DCE applies techniques that you may have learned from working with applications for single machines or other distributed systems. These techniques enable system administrators to manage DCE without having to know about system internals. You can start with a configuration that is appropriate for your initial needs and grow to larger configurations without sacrificing reliability or flexibility. DCE supports large networks with many users, as well as smaller networks.

The following concepts, which are described in the remaining sections of this chapter, are central to DCE system administration:

- Clients and servers to make and respond to requests for a service
- Remote Procedure Calls (RPCs) for client-to-server communications
- Cells, which are groups of users, servers, and machines that share security, administrative, and naming boundaries
- A single namespace that lets client applications identify, locate, and manage objects, including users, machines, servers, groups of servers, and directories
- A single filespace that allows data sharing among users and machines with proper authorization
- Principals, which are entities—including users, servers, and computers—that are capable of communicating securely with other entities
- Access Control Lists (ACLs) to control access to objects
- Caching, which is the technique of using a local copy of information to avoid looking up the centrally stored information each time it is needed
- Replication, which is the process by which copies of information are created and kept consistent

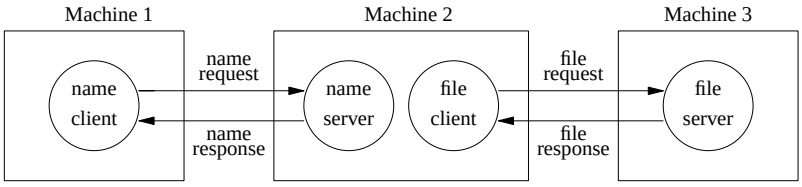
1.1 Clients and Servers

DCE is based on the client/server model. A server is a machine or process that provides a specialized service to other machines or processes. A client is a machine or process that uses a server's specialized service during the course of its own work. Distributed applications consist of a client side that initiates a request for service, and a server side that receives and executes that request, and returns any results to the client. For example, a client can request that a file be printed, and the server where the printer resides carries out that request.

More than one server process can reside on a single machine. Also, one machine can be both a client and a server. For example, a machine can be a client for one DCE component and a server for another.

Figure 1-1 shows a machine that is a name server for a client that issues a name request. The same machine is a client for a file server.

Figure 1–1. Interaction of Clients and Servers



1.2 Remote Procedure Call

A Remote Procedure Call (RPC) is a synchronous request and response between a local calling program and a remote procedure. An RPC begins with a request from a local calling program to use a remote procedure. It completes when the calling program receives all the results (or an error status or exception) from the procedure.

1.3 The Cell

The cell is the basic unit of administration in DCE. A cell usually consists of users, machines, and resources that share a common purpose and a greater level of trust with each other than with users, machines, and resources outside of the cell. Members of a cell are usually located in a common geographic area, but they can also be located in different buildings, different cities, or even different countries, provided they are adequately connected. A cell's size can range from only one machine to several thousand, depending on the size of the organization. All machines in an organization can be included in one cell, or you can choose to have numerous cells within one organization.

Cells designate security, administrative, and naming boundaries for users and resources. Each cell has a name. Cell names are established during the installation and configuration of DCE components.

Members of an organization who are working on the same project are likely to belong to the same cell. For example, in a large organization with several cells, the sales team could belong to one cell, the engineers working on Project X could belong to a second cell, and the engineers working on Project Y could belong to a third cell. On

the other hand, a small organization may have only one cell for both the sales force and the engineers because they share the same level of security and the organization's small size does not warrant the additional administrative overhead that maintaining additional cells requires.

DCE services are managed within the context of a cell, as described by the following examples:

- Each DCE cell typically consists of at least one Cell Directory Service (CDS) server, three Distributed Time Service (DTS) servers, and one Security Service server, as well as the databases that the CDS and Security servers use.
- Pathnames of DCE objects managed by DCE services can be expressed relative to the cell where the objects reside.
- DTS has both local and global servers. Local servers operate within a Local Area Network (LAN). Global servers provide time services anywhere within the cell.

1.4 The Namespace

The namespace is the hierarchical set of names of DCE objects. The top levels of the hierarchy are managed by the Directory Service. Some DCE services (currently the Security Service and Distributed File Service (DFS)) manage their own portions of the namespace. Each DCE object in the namespace consists of a name with associated *attributes* (pieces of information) that describe its characteristics. These objects include resources such as machines or applications.

The namespace contains global namespaces and cell namespaces. A *cell namespace* includes objects that are registered within a cell. A logical picture of a cell namespace is a hierarchical tree with the cell root directory at the top and one or more levels of directories containing names beneath the cell root. The cell namespace is managed by the Cell Directory Service (CDS) component of the Directory Service. Conversely, the *global namespace*, as seen from a local DCE cell, contains objects that are registered outside the local cell, such as the names of other cells. The Global Directory Service (GDS) component of the Directory Service manages one part of the global namespace; a non-DCE service called the Domain Name System (DNS) manages another part of the global namespace.

Administrative tools use the namespace to store information and to locate DCE services. DCE services advertise their locations to the namespace. The namespace provides a means of organizing DCE services into manageable groups.

1.5 The Filespace

Part of the cell namespace is the filespace, which consists of files and directories. These can be physically stored on many different machines, but are available to users on every machine, as long as those users have the proper authorization. You manage the filespace in units called *filesets*, which are hierarchical groupings of related files. Although files are distributed throughout the network, located on and managed by different servers, users see a single filespace. DCE provides administrative tools to assist you in backing up, moving, and replicating filesets.

1.6 Principals

A DCE principal is an identity that is authenticated by the Security Service. When you log into your system, you use your principal name. Principals can be organized into groups and into organizations that contain groups of principals. Information associated with a principal includes information that is traditionally kept in UNIX group and password files, such as the username, group ID, members of a group, and a user's home directory. By default, a principal is known within the bounds of a cell. By creating a special account that indicates you trust another cell's authentication service, you can enable principals from other cells to participate securely within your cell.

1.7 Access Control Lists

An Access Control List (ACL) is an authorization mechanism that allows you to assign permissions that control access to DCE objects. The following DCE objects are protected by ACLs:

- Principals and groups of principals managed by the Security Service
- Files and file system directories managed by the DFS

- DTS servers
- CDS directories and entries
- CDS clients and servers, which have ACLs restricting the use of their management operations (for example, creating a clearinghouse)
- GDS entries managed by GDS's own ACL mechanism, as described in the *OSF DCE GDS Administration Guide and Reference*

An ACL consists of multiple *ACL entries* that define the following:

- Who can use an object
- What operations can be performed on the object

In the filespace, ACLs are an extension of the UNIX system's file protection model. Whereas UNIX file system permissions are limited to the protection of files and directories, DCE ACLs can also control access to other objects, such as individual database entries, objects registered in the cell namespace, and objects managed by applications.

1.8 Caching

Information acquired over the network (for example, through the use of DCE RPC) can be stored in a memory or disk cache on the local machine. This technique reduces network load and speeds up lookups of frequently needed data. For example, information about the namespace and the filespace is cached by DCE client machines.

Caching can be configured on a service-by-service basis. Different caching mechanisms are used for different components of DCE. Each component has configurable options to improve the performance of your installation.

1.9 Replication

Replication increases the availability of resources by having copies of the resource on several machines. For example, with replication you can make database updates on one machine and have them automatically made on other machines in the network.

You can replicate data, move replicas, and control the frequency of updates. The Security Service, CDS, GDS, and DFS all provide replication facilities that are customized for their particular applications.

Chapter 2

Global and Cell Considerations

The purpose of Chapters 2 through 5 is to assist you in planning for the installation, configuration, and maintenance of DCE. For detailed information about installing the DCE source tape and building DCE, refer to the *OSF DCE Release Notes* and the *OSF DCE Porting and Testing Guide*. Part 2 of this guide describes the configuration process, including installing executable files, setting up a DCE cell, and configuring servers and clients.

This chapter discusses how to establish a DCE cell name. This chapter also describes how the cell namespace is organized and provides guidelines for maintaining security and replicating parts of the cell namespace. The last portion of this chapter discusses what you need to consider as you plan for including DFS in your cell.

You need to answer a number of questions when planning for a distributed system. Your answers to these questions determine the basic requirements of your user environment. Keep in mind the following global considerations as you plan for DCE:

- How much do you think your environment will grow in the next few years? Do you anticipate rapid or relatively slow expansion of your network?

  If you think your environment will grow rapidly, consider setting up several cells representing smaller units of your organization. You can manage these smaller units as your network expands. As explained in the *Introduction to OSF DCE*, members of each cell share a common purpose, and the cell is a unit of administration and security. If you anticipate slow expansion of your network, you may be able to establish one or more cells based on the organization that exists now. Consider how many administrators you will need to maintain your DCE cell, based on anticipated future growth.

- How much information does your environment have that needs to be distributed? How much do the users in your network share information?

  If there is a large volume of information that needs to be shared within your network, consider the amount of disk space you require and the number of DFS File Server machines you need.

- How much information updating do you require? Do the users in your network mainly look up information, or do they create and change information at their workstations?

  If information changes frequently and users in your network depend on the accuracy of that information, you need to consider how much you rely on replication. It is better to go to a central source of information for data that changes frequently. If users look up information but do not need to change the information that is shared with other users, you can rely more on replicated data.

- Is the most important data the most available data? Have you made plans to replicate this data?

  CDS, GDS, the Security Service, and DFS maintain master copies of their respective databases. Each CDS directory can be replicated separately. In addition to DFS databases, individual DFS filesets or groups of filesets can be replicated. GDS replication, also known as shadowing, can be done for a single object or an object and its subordinates (a subtree). The Security Service replicates the entire registry database. Because other components depend on the information managed by the Security Service and parts of the CDS namespace, that data needs to be available at all times. For example, the special character string **/.:** (the cell root) is stored in CDS and must always be available.

Keep in mind that while replicating data improves availability, there is a cost in terms of performance and the amount of administration required.

- If your network has a gateway, are the servers located on the same side of the gateway as the clients that rely on those servers?

  CDS servers broadcast messages at regular intervals to advertise their existence to CDS clerks in the network. Clerks learn about servers by listening for these advertisements. Placing the servers and the clients that rely on them on the same side of the gateway facilitates efficient updates of information and a quick response to client requests. Additional administration is required if you rely on servers that are not available through the advertisement protocol, which is effective only in a local area network.

  Consider how fast and how expensive links are if you are administering a cell that includes users in different geographic locations. You may want to keep more information locally to reduce your dependence on transmitting information across links.

- Is communication limited to your own cell, or do you need to communicate with other cells?

  DCE offers two methods of connecting cells so that they can communicate:

  — The standard intercell connection, in which a cell is registered in a global directory service that DCE supports and communicates with other cells registered in that directory service. A cell can be registered in both the GDS and DNS directory services. In this case, the cell has two names: one in GDS format, and one in DNS format. These names are the cell's aliases.

  — The hierarchical cell connection, in which a cell is registered in another cell's CDS namespace, and communicates with other cells also registered in that cell's namespace. In a hierarchical cell configuration, the cell at the top of the hierarchy must be registered with a global directory service, but the cells beneath it do not need to be, because they communicate through CDS.

  Regardless of which method you choose, in order for your cell to communicate with other cells, you must

  — Establish a unique name for your cell and define it in the appropriate namespace (GDS, DNS, or CDS)

  — Have at least one GDA running in the cell

— Establish a Security Service trust relationship with the other cells with which you wish to communicate

2.1 Establishing a Cell Name

You must establish a name for your cell before you configure it. A uniquely identified cell name is critical to the operation of the Security Service; this name is the basis for authentication in your cell. Whether or not your cell name needs to be globally unique depends on your plans for communication with other cells.

If you plan to create a private cell and never intend for it to communicate with cells outside your organization, you are not required to obtain a globally unique cell name. However, if you plan to communicate with the cells of other organizations, you must obtain a globally unique cell name for your cell before you configure it.

If you plan to communicate with other cells through GDS, DNS, or CDS, you must obtain a globally unique name for your cell. The next sections describe how to establish GDS, DNS, and hierarchical names for your cell. See Appendix A of the *OSF DCE Administration Guide—Core Components* for a description of the valid characters supported in GDS, DNS, and CDS.

In some cases, you may need to change the name of your cell after you have configured it, for example, because your company has reorganized and the cell name you established at configuration time no longer reflects the new organization. In other cases, you may need to add another name for your cell, for example, if you initially registered it in GDS and find that you must also register it in DNS. To add a new name for your cell, use the **dcecp cellalias** task object.

2.1.1 Establishing a GDS Cell Name

If you plan to use GDS to communicate with other cells, you must obtain a globally unique name for your cell from the GDS global naming authorities before you configure your cell, then define it in the GDS namespace. The name you obtain for your cell will be in GDS syntax.

As explained in the *Introduction to OSF DCE*, GDS-style names consist of a series of attribute/value pairs, separated by equal signs (=). Each attribute/value pair is called a relative distinguished name (RDN). The directory information tree (DIT) determines the hierarchy of a GDS name; that is, how the RDNs are ordered to create a global name. An example of a GDS-style global name, called a Distinguished Name (DN), is **/.../C=US/O=ABC/OU=DCE/CN=gunther**. Each RDN is separated by slashes (/). See the *OSF DCE GDS Administration Guide and Reference* for a complete description of the GDS naming structure.

DCE cell name values are generally stored in the Organization (**O=**) or Organization Unit (**OU=**) attributes of a GDS name. For example, the global name

/.../C=US/O=ABC/OU=Seattle

uses the **OU=** attribute to store the cell name **Seattle**. There is a fixed set of two-letter codes that must be used to indicate the Country (**C=**) attribute for your cell; for example, **C=US** or **C=JP**. The country in which you reside may also have standard Organizations (**O=**), where your organization is registered as a code. Your organization may also have conventions that apply to the way an Organization Unit (**OU=**) is represented. Check with your naming authority for exact conventions. Any valid X.500 name, including names provided by other standards supported by X.500, can be used as a cell name. Refer to the *OSF DCE GDS Administration Guide and Reference* for details about naming rules, including valid characters, restrictions, and maximum name sizes for GDS names.

To obtain a unique GDS name for a cell, contact the administrator in charge of the portion of the DIT under which you want to name your cell. For example, in the United States, the American National Standards Institute (ANSI) delegates X.500 names that are subordinate to the RDN **/C=US**. Suppose you are an employee of ABC, a U.S. corporation interested in participating in a worldwide X.500 directory. If you wanted to configure a single cell whose name is **/C=US/O=ABC**, you would contact ANSI to reserve ABC as a unique organization name. Similarly, if you wanted to configure multiple cells in your organization and name them based on organization units, you would contact a naming authority within your company to establish a cell entry such as **/C=US/O=ABC/OU=Sales**.

Send X.500 name registration requests to

American National Standards Institute
11 West 42nd Street

New York, NY 10036
Telephone Number: (212) 642-4976

It is the responsibility of the person making the request to ANSI to be sure that your organization does not send more than one request for an organization name. Once you receive your organization name, it is recommended that your organization set up a central administrative authority to manage names that are subordinate to the organization name.

After you have configured your cell, you need to define it in the GDS global namespace. GDS stores cell information in the **CDS-Cell** and **CDS-Replica** attributes. You must add these two attributes to an existing GDS entry for that entry to become a cell entry.

2.1.2 Establishing a DNS Cell Name

DCE also supports global directory operations through the use of DNS. If you plan to use DNS to communicate with other cells, you need to obtain a globally unique name for your cell from the DNS global naming authorities before you configure your cell, then define it in the DNS namespace. The name you obtain for your cell will be in DNS syntax. An example of a DNS-style cell name is:

/.../seattle.abc.com

If you plan to use DNS as your global directory service, your DCE cell name must follow the ARPA Internet Domain System conventions for site names. If you are already an Internet site, you can create one or more cells subordinate to your Internet domain name, depending on how your site is organized. The following conventions govern an Internet-style name:

- The name needs to have at least two levels; for example, **abc.com** or **sctech.edu**. The names in the first two levels are registered with the Network Information Center (NIC), which is the naming authority for DNS names.
- The name cannot be longer than 255 characters.
- The name can contain any number of fields in addition to the two required levels, which are conventionally separated by periods.

- The name needs to end in a suffix that indicates a kind of institution. This last field is the most significant one, in contrast to a GDS name, which begins with the most significant field. The standard suffixes are as follows:
  - **.com** for businesses and other commercial organizations
  - **.org** for noncommercial organizations
  - **.edu** for educational institutions
  - **.gov** for government institutions
  - **.mil** for military institutions
  - **.net** for network support organizations
  - *.xx* for two-letter country codes (such as **.de** for Germany and **.fr** for France) that conform to the International Organization for Standardization (ISO)

Refer to the *OSF DCE Administration Guide—Core Components* for further information about naming rules, including valid characters, restrictions, metacharacters, and maximum name sizes for DNS names.

To obtain a unique DNS name, contact the administrator in charge of the subtree under which you want to name your cell. Send registration requests to the NIC at the following Internet address, telephone number, FAX number, or mailing address:

HOSTMASTER@NIC.DDN.MIL

Telephone Number: (800) 365-3642 between the hours of 7:00 a.m. and 7:00 p.m. Eastern Standard Time

FAX (703) 802-8376

Government Systems, Inc.
Attention: Network Information Center (NIC)
14200 Park Meadow Drive
Suite 200
Chantilly, VA 22021

After you have configured your cell, you need to define it in the DNS global namespace by a creating cell entry for it in DNS. To create a cell entry in DNS, an administrator must edit a data file that contains *resource records*.

You also need to establish cross-cell authentication with any other cells with which you want to communicate.

2.1.3 Establishing a Hierarchical Cell Name

Cells in a hierarchy use CDS to communicate. Consequently, if you plan to add your cell to an existing cell hierarchy, you establish a globally unique name for it by creating a CDS name that represents your cell in the CDS namespace of another cell, then appending this CDS name to the global name of the cell in which you created your cell's CDS name. (A cell's global name is its name starting from the **/...** global root prefix.) The cell whose global name you use, and in which you have created a CDS name for your cell, is known as your parent cell, while your cell is known as a child cell. For example, if your parent cell's global name is:

/.../C=US/O=ZOMBIE/OU=WOOF

and the CDS name you created for your cell is **zappa**, the name of your cell is:

/.../C=US/O=ZOMBIE/OU=WOOF/zappa

Note that in this situation, the parent cell's global name may contain CDS syntax as well as GDS or DNS syntax, depending on where it exists in the hierarchy. For example, your parent cell's global name could be

/.../coolco.com/sales/northeast

where **/.../coolco.com** is the DNS namespace portion of the name, in DNS syntax, and **/sales/northeast** is the CDS namespace portion of the name, in CDS syntax. (Global names are also referred to as fully qualified names. Both terms refer to names that begin at the global root directory **...**.)

Note that you currently cannot use the DCE configuration program to configure a cell as a child cell for addition into a cell hierarchy; the cell must already have been configured before you can make it a child cell in a hierarchy. See Chapter 21 of the

OSF DCE Administration Guide—Core Components for instructions on how to create a child cell.

2.2 The Cell Namespace

An integral part of planning for a DCE cell is understanding the organization of your cell namespace. Consider the following as you plan the organization of a cell in your network:

- Are security requirements maintained?
- Does the organization of the cell facilitate network traffic where data sharing needs are the greatest?
- How will you manage the administrative accounts created for each DCE service during the configuration process?
- What are your DFS administrative domains (groups of DFS servers that are administered as a unit)? Can you group servers for more efficient administration?

2.2.1 Determining Cell Boundaries

In DCE, the boundaries of a cell are equivalent to the boundaries of the cell namespace. A small organization can consist of one cell. A large organization can have many cells. The primary factors in determining a cell's boundaries are the common purpose and trust shared by the cell's principals. Principals within a cell can belong to groups that share the same privileges. Members of a group share the same level of trust and are authorized to perform certain actions.

Because there is a set of administrative tasks associated with setting up and maintaining each cell, it is reasonable to keep the number of cells in your organization to a minimum. However, the level of trust shared by groups of principals is a more important consideration than administrative overhead.

2.2.2 Keeping Cells Stable

Once you decide how many cells you need and where the boundaries of those cells will be, make an effort to keep your cell structure stable. Servers are not easily moved from one host to another, so be sure to plan your namespace structure carefully in order to minimize reconfiguration. If you do need to move a machine from one cell to another, you must do the following:

- Use the **dce_config REMOVE** command to remove all configuration data from the machine (if the machine is a client machine, use the **dce_config UNCONFIGURE** command to remove the client from the cell before using **REMOVE**).
- Reconfigure the host in the new cell.
- Delete any namespace or registry entries for the host in the old cell.

2.2.3 Types of Cell Namespace Entries

The following subsections describe the different types of entries that comprise the cell namespace. These entries are created when you follow the default configuration path described in Part 2. The *OSF DCE Administration Guide—Core Components*, the *OSF DCE GDS Administration Guide and Reference*, and the *OSF DCE DFS Administration Guide and Reference* provide details about the names that the DCE components use. The cell namespace can be divided into three major parts:

- The CDS part of the namespace
- The Security part of the namespace
- The DFS part of the namespace (the filespace)
- The **dced** (per host) part of the namespace

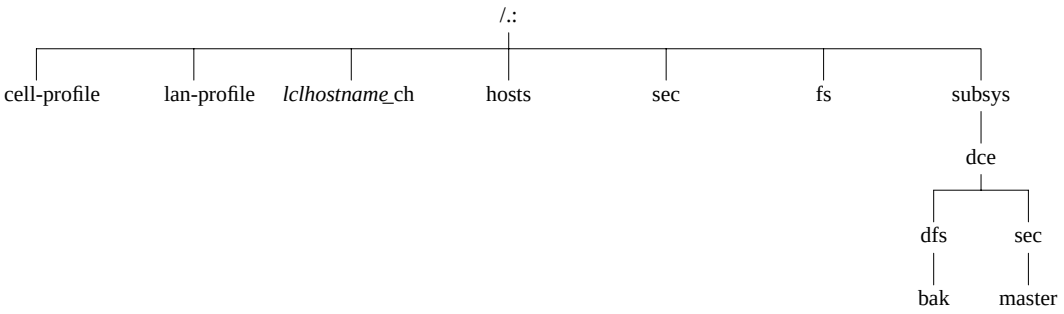
Each of the DCE services maintains its own namespace within the cell namespace. DFS maintains its own namespace to ensure consistency among many files. The Security Service maintains its own namespace to ensure that the DCE cell remains secure. Clients of these two services query CDS for binding information that enables them to find Security or DFS servers. The points where the binding information is stored serve as mount points in the CDS namespace for the namespaces that DFS and the Security Service manage. This transition point between two namespaces is called

a *junction*. The **/.:/sec** directory is the junction from the CDS part to the Security part of the cell namespace, and the **/.:/fs** directory is the junction from the CDS part to the DFS part of the cell namespace.

The junction **/.:/hosts/***hostname***/config** is the junction from CDS to the **dced** (per host) part of the namespace.

Figure 2-1 shows the top level of the cell namespace. In some cases the names in the cell namespace are fixed (or well known) and cannot be changed. In other cases you can choose a different name from the one listed. For more information about which names are well known, see the *OSF DCE Administration Guide—Core Components*. In Figure 2-1, **/.:** and **cell-profile** are well-known names.

Figure 2–1. Top Level of the Cell Namespace



You can use the CDS browser (**cdsbrowser**) or the DCE control program (**dcecp**) to view the CDS part of the namespace, including the **sec** and **fs** junctions. You can use commands such as **ls** to see the contents of the DFS part of the namespace and **dcecp** to see the contents of the Security portion.

2.2.3.1 CDS Namespace Entries

The CDS namespace contains entries for servers, hosts, CDS clearinghouses (collections of directory replicas stored at a particular server), RPC profiles, RPC groups, and subsystems. The entries have a CDS type of **directory** or *object*, indicating the kind of CDS object to which the name refers. A third CDS type,

softlink, is an entry that points to another entry. A CDS directory is a container in which objects are stored. CDS uses directories to organize groups of object entries.

In addition, the CDS namespace provides specialized services for other DCE components, such as location information contained in the fileset location database (FLDB), which is the database that maps filesets to the file server machines on which they reside.

Profiles cataloged in the CDS namespace specify a search path through the Directory Service. The cell profile (**/.:/cell-profile**) stores the location of the servers that are available in the cell, regardless of physical location. In a geographically dispersed cell, servers can be located in different cities or even different countries. The LAN profile defines alternate servers that can be used in situations where geographic proximity is important. For example, **/.:/lan-profile** is the default LAN profile used by DTS. This profile contains entries for the DTS server local set. If a cell spans more than one LAN, another layer can be created below **/.:/lan-profile** to specify the location of the profile for each part of the cell. For example, in a cell that encompasses two LANs, you can direct hosts on one LAN to **lanA-profile** and hosts on the other LAN to **lanB-profile**. For information on setting up multiple LAN profiles, see Chapter 8.

2.2.3.2 Security Namespace Entries

The types of Security entries are as follows:

- **principal**

  This type of entry contains an individual principal.

- **principal** directory

  This type of entry contains individual principals or one or more principal directories, or both.

- **group**

  This type of entry contains an individual group.

- **group** directory

This type of entry contains individual groups or one or more group directories, or both.

- **org**

  This type of entry contains an individual organization.

- **org** directory

  This type of entry contains individual organizations or one or more organization directories, or both.

- **policy**

  This type of entry contains Security policy.

When you (or an application) are accessing an entry in the Security part of the namespace, the name of the entry alone provides enough information for the Security Service to work with. For example, the Security server knows that the login name is a principal name that is registered in the Security part of the namespace; **/.:/** *principal_name*, **/...**/*cell_name/principal_name*, and *principal_name* are all valid ways of representing the name you use to log in.

When you use **dcecp**, you specify the type of object you will operate on before you operate on it. For example, to change account information associated with the principal **smith**, you specify that you want to operate on a principal, and you then enter the principal name **smith**. **dcecp** deals with the following three types of objects:

- Principals
- Groups
- Organizations

The *OSF DCE Administration Guide—Core Components* explains how to use **dcecp** to display information related to principals, groups, organizations, and accounts.

In addition to objects registered in the Security space, **dcecp** operates on all objects in the namespace. To operate on these objects, **dcecp** requires the object's fully qualified pathname, as shown in the following example:

/.:/sec/principal/smith

and not simply the following:

smith

The following parts of the namespace comprise the Security namespace:

- **/.:/sec/principal**
- **/.:/sec/group**
- **/.:/sec/org**
- **/.:/sec/policy**

2.2.4 CDS Namespace Replication Considerations

Directory replication is the most reliable way to back up the information in your CDS namespace. Because the CDS data is replicated by directory, when you replicate a directory, all of the entries in it are automatically replicated. Use **dcecp** to create replicas of directories at a CDS clearinghouse. Clearinghouses need to be created in the root directory (**/.:**) of the cell namespace.

Follow these guidelines for replicating parts of the cell namespace:

- The root (**/.:**) is automatically replicated when you create a clearinghouse.
- You should have at least two copies of each CDS directory to ensure the entire namespace is available at all times. For further information about backing up CDS information, see the *OSF DCE Administration Guide—Core Components*.

2.3 Planning for Access Control

When planning for access control, it is important to keep the level of access control in your cell restrictive enough to ensure that security is maintained. A special set of individuals or a special group can be given permission to create accounts and groups in the root directory of the Security space. A group called **acct-admin** is created when you configure DCE. The **acct-admin** group is the only group that can create accounts and groups in the root directory of the Security space.

While maintaining an adequate level of security in your cell, you also need to consider the requirements of administrators who are maintaining DCE services when you set access control levels. For example, if one person is responsible for administration of DFS in your cell, that person may need to add servers to the Security and CDS namespaces. On the other hand, an administrator responsible for the Security Service manages the Security server but does not control the DFS filespace.

Following are some of the groups created when you configure DCE using the DCE configuration script:

- **sec-admin**

  This group administers Security servers, registry replication, and other Security functions.

- **cds-admin**

  This group administers CDS servers, CDS replication, and other CDS functions.

- **dts-admin**

  This group administers DTS servers and related DTS functions.

- **dfs-admin**

  This group administers DFS file servers and related DFS functions.

- **audit-admin**

  This group administers the Audit daemon and related Audit Service functions.

See Section A.2.3 for a list of DCE groups created by the DCE configuration script.

In addition to the administrative groups, individual users need permission to control some information kept in the registry database. For example, a user needs to be able to change her or his password, home directory, or login shell.

2.4 The Filespace

The following subsections contain guidelines for planning your cell's filespace. The *OSF DCE DFS Administration Guide and Reference* explains some of these planning considerations in more detail.

The filespace begins under the cell root at the **/.:/fs** junction to DFS from the CDS namespace. The notation **/:** is set up by default to be equivalent to **/.:/fs**. Thus, the notation **/:/usr/***user_name* is equivalent to **/.:/fs/usr/***user_name*.

Some parts of DFS run in the host machine's kernel. This kernel function must be present on your machine before you run DFS.

2.4.1 DFS Administrative Domains

A *DFS administrative domain* is a collection of machines in the same cell that are configured for administration as a single unit. In a single cell you can have one or many administrative domains, depending on the size of your organization. Organizing DFS server machines into different administrative domains simplifies the management of the cell filespace by creating smaller units for administration. All machines within an administrative domain must be in the same cell.

2.4.2 DFS Administrative Lists

DFS administrative lists are files that define the principals and groups that can perform actions affecting specific server processes on a server machine. There is one DFS administrative list for each DFS server process running on a machine. For example, a server's **admin.bos** file defines who has administrative rights to the BOS server (**bosserver**), and thus determines who can manipulate and maintain server processes on that one server. Groups, as well as individual users, can be placed on an administrative list. Each server machine stores administrative lists for its processes on its local disk. A process automatically creates its initial administrative list when it is started if the list does not already exist on the local disk of the machine.

2.4.3 Determining the Roles of DFS Machines

Follow the recommendations in the the *OSF DCE DFS Administration Guide and Reference* when you assign roles to the DFS machines in your cell.

The first DFS machine that you configure during DCE installation and configuration (described in Chapter 9) needs to function as a *System Control Machine*. The System Control Machine is the server that distributes DFS configuration information. Next you configure a *Fileset Location Database Server*, which is the server that maintains the fileset location database. The DCE installation and configuration script assumes that the **root.dfs** fileset, which is the fileset that corresponds to the top (**/.:/fs**) level of the file tree, is located on the Fileset Location Database Server. Section 2.4.5 and the *OSF DCE DFS Administration Guide and Reference* contain further information about **root.dfs**.

Machines that you configure as DFS servers can run the processes required to be DFS File Servers. Be sure the machine you choose has enough space to store LFS filesets. The amount of free space you need depends on how much data you plan to store in LFS filesets. Filesets on File Servers can store DFS client binaries in addition to user files. These filesets can also be distributed on other file server machines in your cell. In addition, if your domain has only one server machine, this machine must run all processes and fill all required machine roles. For example, in addition to being a System Control Machine, this machine must be a File Server and a Fileset Location Database Server. If your domain has three or more DFS server machines, three machines need to store DFS databases. An odd number of DFS database machines is recommended.

2.4.4 Setting Up the DFS File Tree

Follow the recommended conventions in this section when you set up your DFS file tree. (For more information about this process, see the *OSF DCE DFS Administration Guide and Reference*.

Below **/.:/fs** are directories that help organize your DFS environment, such as

- The **common** directory

This directory contains programs and files needed by users working on machines of all system types, such as text editors or online documentation files. The **common/etc** directory is a logical place to keep the central update sources for files used on all DFS client machines.

- The **public** directory

  This directory contains files that users want to make available to everyone, including foreign and unauthenticated users.

- The **sys_type** directory

  This directory contains binaries for each system type you use as a file server or client machine. If you plan to use the **@sys** variable in pathnames, you need to use standard names to represent system types.

- The **usr** directory

  This directory contains the home directory of each DFS user in a cell and any foreign users that are granted a local account. Users and system administrators can protect this directory so that only locally authorized users can access it. If your cell is quite large, you can divide user home directories in multiple directory listings to facilitate quicker directory lookups.

- The **src** directory

  This directory contains source filesets, such as those for DFS source files.

2.4.5 Setting Up Filesets

Consider the following recommendations and restrictions when you set up filesets:

- Fileset names must be limited to 102 characters or less.
- Every cell must include **root.dfs**. The root fileset can be a LFS fileset or it can be a non-LFS fileset (a non-DCE LFS file system). If **root.dfs** is a LFS fileset and you plan to use replication, you need to follow the steps described in the *OSF DCE DFS Administration Guide and Reference*, which describes how to create **root.dfs** as a DFS LFS fileset and create a read/write mount point for the fileset below the top level of the cell's filespace.

- You should use a common prefix when naming related filesets. This aids in manipulating and grouping related filesets. It also relates the fileset's name to its mount point.
- You can group filesets on the same partition of a File Server machine. This can localize the effects of an outage, but you also need to consider factors such as the number of File Server machines and load balancing before grouping filesets.
- You can replicate filesets for load balancing and to make fileset contents more available. Replication is appropriate for filesets that are read much more often than they are written, such as filesets containing installed executable files. Replication is not supported for non-LFS filesets.
- Consider the disk space a fileset requires before setting up filesets.

2.4.6 Using @sys and @host Variables

Follow the suggested conventions in the *OSF DCE DFS Administration Guide and Reference* when using the **@sys** and **@host** variables in certain pathnames. When the DFS *Cache Manager* encounters one of these variables, it substitutes a string that consists of the local machine's architecture and operating system type for **@sys** or the hostname for **@host**, causing a certain directory to be used. Using **@sys** and **@host** is helpful when you are constructing symbolic links from the local disk to DFS. You can create identical symbolic links on all machines, but each machine transparently accesses the files appropriate to its system name or hostname. The DFS **cm sysname** command sets and displays the current value for **@sys**.

Chapter 3

Client and Server Considerations

This chapter describes configurations for DCE client machines, the different types of DCE server machines, DCE remote administration machines, and DCE Application Development Environment machines. A DCE client machine can run client code of every DCE service. DCE server machines are configured to run a certain set of the DCE software. A DCE server software package is made up of at least one daemon and, in some cases, one or more additional programs that comprise the server side of the DCE component. DCE server machines also run the DCE client software. DCE remote administration machines, which are client machines specially configured for remote server administration, contain certain administration programs in addition to the DCE client software. The DCE Application Development Environment configuration contains files (such as header files) needed by DCE application programmers, in addition to the DCE client software.

When planning the configuration of your DCE clients and servers, remember space needs. A machine that has a particular configuration of the DCE software will need enough space for both the DCE software and the operating system software. See the *OSF DCE Release Notes* for detailed information on space requirements for the various DCE machine configurations.

The sections of this chapter are presented in the order in which you need to approach configuring DCE machines.

3.1 Requirements for DCE Client Machines

The following subsections describe the requirements for setting up DCE client machines. They also discuss some considerations for configuring DCE clients. Remember that all DCE machines, including DCE server machines, are DCE clients. Therefore, be sure to add the appropriate server space requirements to the DCE client space requirements to reach an approximate total space requirement for the client machine.

3.1.1 Files Installed on DCE Client Machines

This section gives an overview of the software that is installed on DCE clients. Additional details are provided in Sections 3.1.2 through 3.1.7.

3.1.1.1 Minimum DCE Client

A minimum DCE client configuration contains client services for DCE RPC, CDS, Security, and DTS.

3.1.1.2 Full DCE Client

In addition to the files needed for the minimum DCE client configuration, the full DCE client contains client files for GDS and DFS. The files for a full DCE client are as follows; this list includes the files needed for a minimum DCE client:

- Daemons

  auditd, **cdsadv**, **cdsclerk**, **dced**, **dtsd**

- Utilities

Note: **dcecp** replaces **acl_edit**, **cdscp**, **dtscp**, **rgy_edit**, and **rpccp**. Although these utilities may still be in use, and are still described in the documentation, they are no longer supported. In addition, they do not offer the most recent and complete functionality that **dcecp** offers.

dcecp, **dce_login**, **getcellname**, **gdscp**, **kdestroy**, **kinit**, **klist**, **uuidgen**

- Data Files and Shell Scripts

  cds_attributes, **cdscache.schmid**, **cds_globalnames**, **cdscp.bpt**, **cdscp.mbf**, **dce_config**, **dce_shutdown**, **dce.clean**, **dce.rm**, **dtscp.bpt**, **localtime**, **posixrules**, **pwd_config**, **rc.dce**, **rc.dfs**, **zoneinfo**

- Libraries

  libdce, **libdcecp**

- Message Catalog Files

  dcecds.cat, **dcedcp.cat**, **dcecfg.cat**, **dcedts.cat**, **dceevt.cat**, **dcekdb.cat**, **dcekdc.cat**, **dcekrb.cat**, **dcerpc.cat**, **dcesad.cat**, **dcesec.cat**, **dcethreads.cat**, **gdsditadm.cat**, **gdsproc.cat**, **gdssysadm.cat**, **idl.cat**, **uuidgen.cat**

- DFS Message Catalog Files

  dfsasy.cat, **dfsbak.cat**, **dfsbbs.cat**, **dfsbdb.cat**, **dfsbtc.cat**, **dfsbtm.cat**, **dfscmd.cat**, **dfscmp.cat**, **dfsdau.cat**, **dfsdcl.cat**, **dfsepi.cat**, **dfsfsh.cat**, **dfsfts.cat**, **dfshst.cat**, **dfslgb.cat**, **dfssal.cat**, **dfstkm.cat**, **dfsubk.cat**, **dfsupd.cat**, **dfsvls.cat**, **dfsxcr.cat**, **dfsxvl.cat**

- Files on AIX Only

  The following files are needed on the AIX platform only; they are kernel extensions. On the OSF/1 platform, they are already linked into the kernel.

  config_kern_ext, **dtskernext**, **dtsloadobj**, **load_kern_ext**, **query_kern_ext**, **unload_kern_ext**

- DFS Clients Only

  The following files are needed only if the DCE client machine is also a DFS client machine:

dfsbind, **dfsd**

The following are optional on a DFS client:

bos, **cm**, **fts**

- AIX DFS Clients Only

  cfgdfs, **cfgexport**, **dfscmfx.ext**, **dfscore.ext**, **dfsloadobj, export.ext**

The following subsections describe the executables that run on a DCE client machine.

3.1.2 RPC Client Programs

A DCE client contains the following programs:

- The **dced** daemon must run on any machine that has a DCE RPC server process that exports an interface with dynamic bindings. The **dced** daemon is used to register binding information (among other things). The **dced** daemon must run on every DCE machine because on every DCE machine there are client-side daemons that export interfaces. For example, the **dtsd** daemon exports the **acl** interface.

  The **dced** daemon must be running before you configure any other DCE services because DCE services need to register their endpoints with **dced**. Only one **dced** daemon is needed on a machine. In fact, only one can run on a machine at a time because it uses a well-known port.

  Network interfaces, routing services, and other network services must be available before DCE RPC starts. The **dced** daemon is started in the **/etc/rc.dce** file. The **/etc/rc.dce** file can be invoked by other **rc** files, such as **/etc/rc** and **/etc/rc.local**, so that DCE services can be brought up each time a machine boots.

- The DCE control program (**dcecp**) for the management and maintenance of the DCE RPC software. Section 3.3 describes **dcecp**.

3.1.3 Security Service Client Programs

Every DCE client machine has, as part of the **dced** daemon, the Security Validation Service. This service takes the place of the machine principal. Most principals are interactive users, but the machine principal is not. The Security Validation Service performs the processing necessary so that other daemon processes on the machine appear to be running with the machine's identity.

The Security Validation Service periodically refreshes the ticket-granting ticket for the machine's principal. A DCE client machine must have a valid ticket-granting ticket in order for a principal to use DCE services. The Security Validation Service also exports the interface that assures a Security client that it is actually contacting the real Security server when the client requests a ticket-granting ticket from the Security server.

3.1.4 Audit Service Client Programs

There are no Audit service client programs. The clients of this service are the server processes of the DCE services that use auditing, for example, the Security Service's **secd** daemon.

3.1.5 CDS Client Programs

The DCE client runs the following CDS processes:

- The CDS advertiser, the **cdsadv** process, does the following:
  - — Allows applications to locate and communicate with **cdsd** servers
  - — Starts any needed CDS clerks (**cdsclerk**)
  - — Creates the cache shared by local CDS clerks
- The **cdsclerk** is an interface between CDS client applications and CDS servers. A clerk must exist on every machine that runs a CDS client application. One **cdsclerk** process runs for each DCE principal on a machine that accesses CDS. The CDS clerk handles requests from client applications to a server and caches the results returned by the server. Because the results of the server request

are cached, the clerk does not have to go repeatedly to the server for the same information. All CDS clerks on a machine share one cache. One clerk can serve multiple client applications running on the same machine.

3.1.6 DTS Client Programs

The DCE client runs the following DTS processes:

- The **dtsd** daemon is set to be a client or a server. On a client machine, **dtsd** synchronizes the local clock.
- The DCE control program (**dcecp**) for the management and maintenance of the DTS software. Section 3.3 describes **dcecp**.

3.1.7 GDS Client Programs

This section describes the programs that make up the client side of GDS. The DCE configuration script installs the GDS client software but does not configure it.

To configure and activate a GDS client, run the **gdssysadm** program, then initialize the Directory User Agent (DUA) cache by running **gdsditadm**. For details on these programs and on setting up and activating GDS, see the *OSF DCE GDS Administration Guide and Reference*.

The GDS control program (**gdscp**) is a utility that you can use for managing and maintaining the GDS software. For additional information on the **gdscp** program, see the *OSF DCE GDS Administration Guide and Reference*.

If GDS is installed, the DCE client runs the DUA. The DUA, which is the client side of GDS, sends requests to the GDS server process, the DSA. The DUA consists of the following processes:

- The **gdscache** process caches user data and stores data used for regulation purposes locally.
- The **gdscstub** process handles all outgoing requests to remote DSAs.
- The **gdscacheadm** program supports administration of the contents of the local DUA cache database.

- The **gdsipcchk** program verifies the IPC-state information contained in the shared memory area of a GDS installation.
- The **gdssysadm** program supports administration of the local GDS installation, such as configuring GDS, activating servers, and backing up the database.

A machine running only the client side of GDS can access GDS servers on other machines, or one machine can run both the client and server portions of GDS. Machines running just the DUA are known as *client systems*. Client systems can access directory information on server machines without having to store that information.

3.1.8 DFS Client Programs

If DFS is installed, the DCE client runs the following processes:

- The Cache Manager process (**dfsd**) initializes the cache manager in the kernel, alters configuration settings, and starts background daemons.

  The **dfsd** process is responsible for the local caching of file and directory data on machines used as DFS clients. When the **dfsd** process starts, it initializes the cache. When a client retrieves part of a file from a remote File Server, the **dfsd** process keeps a copy of that part of the file on the client machine's local disk. As long as that part of the file does not change, the locally cached copy remains available to the client. A new copy is retrieved from the DFS File Server machine only when another process changes the cached portion of the file. The **dfsd** process also caches directory and fileset location information.

- The **dfsbind** process does the following:
  - — Obtains cell location information from CDS
  - — Responds to Security Service requests on behalf of the DFS kernel processes by making calls to the Security server

3.2 Requirements for DCE Server Machines

The following subsections describe the considerations involved in setting up DCE server machines.

3.2.1 Files Installed on DCE Server Machines

The following subsections list the files that must be installed on each of the different DCE server machines. Remember that because all DCE servers are also DCE clients, the files described in Section 3.1 must also be installed on server machines. Therefore, add the appropriate client space requirements to the DCE server machine space requirements to reach an approximate total space requirement for the server machine.

3.2.2 DCE RPC Server Programs

There are no DCE RPC server programs other than the programs that run on the DCE client.

3.2.3 Security Server Processes

Every cell has one master Security Service machine and can also have slave Security Service machines. The following processes run on a Security Service master or slave server machine:

- The Security server, or **secd** process, implements the Authentication Service, the Privilege Service, and the Registry Service.
- The **sec_create_db** program initializes the Security database. You give this command an option indicating whether you want to create a master or slave Security server on the machine.
- The DCE control program (**dcecp**) for the management and maintenance of the Security software. Section 3.3 describes **dcecp**.

Keep the following considerations in mind when you are planning for Security servers:

- The node that runs the master Security server must be highly available and physically secure. Consider placing the master Security server machine in a locked room and keeping a log to record who accesses the machine.
- Be sure to move the master Security server before removing the node from the network or shutting down the node for an extended period of time. Modifications are made to the master Security server and propagated to slaves throughout your cell. If the master Security server is unavailable, no updates can be made.
- A cell can have only one master Security server. If you plan to make one cell out of several existing cells with independent master Security servers, you must first merge their registries.
- If the host that contains the master Security server goes down, hosts that have slave servers can still provide registry information, so consider having a number of slaves in your network. Use factors such as the number of machines in your cell, the reliability of the machines that run Security servers, and your cell's available resources to determine how many slave Security servers you need to have.

For further information about planning for the Security Service, see Chapter 38 of the *OSF DCE Administration Guide—Core Components*.

3.2.4 Audit Server Processes

An Audit server provides the other DCE services with access to the DCE auditing facilities. An Audit server runs the **auditd** daemon. When auditing is available in a DCE cell, each machine must run the daemon.

3.2.5 CDS and GDA Server Processes

A CDS server stores and maintains object names within a cell and handles requests to create, modify, and look up data. One of the CDS server machines in a cell must be configured as a GDA server as well. There must be a GDA server (the **gdad** daemon) in a cell in order for the cell to communicate with other cells.

The following processes run on a CDS server machine:

- The CDS daemon, **cdsd**, is the CDS server process.
- The **cdsadv** on a DCE client machine, receives server advertisements to find out what servers are available. On a CDS server machine, it also sends server advertisements.
- The DCE control program (**dcecp**) for the management and maintenance of the CDS software. Section 3.3 describes **dcecp**.

When preparing for CDS, you need to select server nodes that store and maintain the clearinghouses (CDS databases) in the cell. Keep the following guidelines in mind in order to achieve reliability, optimum performance, and data availability:

- Choose dependable nodes. A CDS server wants to avoid downtime as much as possible and needs to be restarted quickly when downtime occurs. The CDS server needs to be one of the first systems available on the network because client applications and other DCE servers rely on the CDS server for up-to-date information. The CDS server initializes the CDS namespace when you configure DCE.
- Use reliable network connections. This helps to ensure that all servers maintaining directory replicas can be reached when CDS performs a skulk. Skulks are periodic updates that check for consistency across all replicas.
- Consider the size of your cell and how geographically dispersed the cell is when deciding how many CDS servers you need. You should have at least two copies (one master and one replica) of each CDS directory to ensure access to data if one of the servers becomes unavailable.
- Each CDS server in a cell must maintain at least one clearinghouse. All clearinghouses should contain a copy of the root, in addition to other directories replicated there.
- Make replication decisions based on where the contents of directories are referenced. Put replicas where the contents are read and put masters where the contents are written.

In a DCE configuration that uses GDS or DNS, CDS must be able to contact at least one GDA to access global directory service. CDS contacts the GDA via the **gdad** daemon, which sends lookup requests for cell names to either GDS or DNS and returns the results to the CDS clerk in the cell that initiated the request.

The GDA can be on the same machine as a CDS server, or it can exist independently on another machine. You should have at least two **gdad** daemons running in a cell to ensure GDA availability.

3.2.6 DTS Server Programs

The DCE client configuration already contains all the files necessary for a DTS server machine, with the exception of the optional time provider. The necessary files are as follows:

- The **dtsd** daemon, which is also installed on a DCE client machine, is configured to run as a server when installed on a DTS server machine. As a server process, **dtsd** synchronizes with other DTS servers, in addition to synchronizing the local clock, as it does on a client machine.
- The **dts_***device_name***_provider** specifies the communications between the DTS server process and the time-provider process. For *device_name*, substitute the device you are using, which can be a radio, clock, or modem, or another source of UTC time for DTS. A time provider is optional. If you use a time provider, it must connect to a server process.
- The DCE control program (**dcecp**) for management and maintenance of the DTS software. Section 3.3 describes **dcecp**.

Consider the following guidelines when planning your DTS implementation:

- Each cell needs to have at least three DTS servers. At least three DTS servers are needed in order to detect if one of them is faulty when they are queried for the time. It is preferable to have four or more DTS servers to provide redundancy. The additional servers increase the accuracy of time synchronization. However, increasing the number of servers queried for the time also increases the activity on the network. The administrator must balance the level of accuracy with the amount of network activity.
- A time provider is optional in DTS; however, cells that must be closely synchronized with a time standard need to have at least one time provider.
- Servers need to be located at the sites with the greatest number of different network connections.

There are many network configuration decisions that affect DTS planning. In Chapter 24 of the *OSF DCE Administration Guide—Core Components*, you can find details about the total DTS planning process, including configuration planning for Local Area Networks (LANs), extended LANs, and Wide Area Networks (WANs). The *OSF DCE Administration Guide—Core Components* also explains the criteria you need to use when selecting a time source for your network to use.

3.2.7 GDS Server Programs

A GDS server machine requires the following files:

admscheme, **asn1_attr**, **common**, **countries**, **dirparam**, **gdscache**, **gdscacheadm**, **gdscacheupd**, **gdschdb**, **gdscmxl**, **gdsconf**, **gdscrontab**, **gdscstub**, **gdsdaemon**, **gdsdbread**, **gdsdbwrite**, **gdsdeact**, **gdsdirinfo**, **gdsdistcmd**, **gdsditadm**, **gdsdsa**, **gdsexec**, **gdsgendb**, **gdshdlcache**, **gdshdlupd**, **gdsinfo**, **gdsipcchk**, **gdsipcinit**, **gdsipcstat**, **gdslanguage**, **gdslog**, **gdsmkiss**, **gdsmkupd**, **gdssstub**, **gdsstart**, **gdsstep**, **gdssysadm**, **gdstransfer**, **gdsutil**, **ipcconf**, **newscheme**, **nsapmacros**, **osiforminfo**

A machine that is configured as a GDS server runs the GDS client/server configuration, which consists of the following three parts:

- Server
  - — The **gdsdsa** program is the main DSA program; it forks as many DSA processes as are needed and it accesses the database.
  - — The **gdsstub** program is the process that receives incoming requests from clients and responds back to clients, and sends outgoing requests from servers to other servers and receives responses to these requests.
- Client
  - — An application, such as **gdsditadm**, links the DUA library.
  - — The **gdscacheadm**, **gdscache**, and **gdscstub** programs, which are described in Section 3.1.6, provide additional GDS client functionality.
- Per-machine utilities
  - — These utilities are the **gdsipcchk** and **gdssysadm** programs, which are described in Section 3.1.6.

You can have more than one GDS server (DSA) running in your cell. If you have more than one DSA, the data in the Directory Information Base (DIB), which is the GDS database, can be partitioned by storing a different part of the DIB on each server. Alternatively, the data can be replicated by storing copies of the DIB on several machines. A combination of partitioning and replication can also be used.

You need to plan what information you want to replicate and how the information is distributed. The master DSA is the only place where writes and updates can occur. Although you can access the master DSA from any client machine, if the master is unavailable no updates can be made. Therefore, you need to choose a dependable machine to run the master DSA. By creating shadows (replicas) of the master DSA you increase the reliability of read operations. By strategically placing shadows on the network you can improve access time for users.

Keep the following considerations in mind when planning for a DCE configuration that includes GDS:

- The initial installation takes approximately 20 megabytes. One database entry requires approximately 7 kilobytes. You must consider and allocate additional disk space depending on the amount of information you want to store in your directory.
- You must understand how CDS cell-related information is entered and displayed using the **gsdcp** program. (See the *OSF DCE GDS Administration Guide and Reference* for a description of the masks (menus) you use to enter this information.)
- You must know the IP address, port number (together, these two pieces of information are called the PSAP), and the cell name for each machine that has a DSA. (See the *OSF DCE GDS Administration Guide and Reference* for information about entering and displaying PSAP addresses using the **gdsditadm** or **gdscacheadm** GDS administration program.)

3.2.8 DFS Server Programs

DCE supports configuration of the following types of DFS server machines:

- DFS-private File Server machine
- System Control machine

- File Server machine
- Fileset Location Database (FLDB) machine

The following programs are installed on a basic DFS-private File Server machine:

bos, **bosserver**, **dfsbind**, **dfsexport**, **epimount**, **fts**, **ftserver**, **fxd**, **repserver**, **salvage**

The following program is installed on the basic DFS-private File Server machine, on AIX only:

epidaemon

For the System Control machine, the following program is added:

upserver

For the File Server machine, the following programs are added:

newaggr, **upclient**

For the Fileset Location Database machine, the following programs are added:

flserver, **newaggr**, **upclient**

The following programs are optional for DFS servers:

bak, **bakserver**, **butc**, **cm**, **fms**, **repserver**, **scout**, **upclient**, **upserver**

DFS File Servers can assume different roles. The DFS space requirements may vary, depending on the role of a particular machine. (See the *OSF DCE DFS Administration Guide and Reference* for further information about DFS configuration options.) DFS machines that export data for use in the global namespace can run the following server processes:

- The **flserver** process maintains a complete list of fileset locations in the FLDB. The FLDB is a cell-wide database that maps filesets to the servers on which they are located. There must be at least one **flserver** process running in a cell.
- The **fxd** daemon is a user-space process. The **fxd** daemon starts the kernel processes that implement the File Exporter.

- The **ftserver** process allows system administrators to create, delete, duplicate, move, back up, or restore entire filesets with one set of commands.
- The **bosserver** process reduces system administration demands by constantly monitoring the processes running on its File Server machine. The **bosserver** process can restart failed processes automatically; it provides a convenient interface for administrative tasks.
- The **repserver** process manages replicas of filesets on all File Server machines.
- The **upserver** process controls the distribution of common configuration files to all other DFS server machines in a domain.
- The **upclient** process contacts the **upserver** process to verify that the most recent version of each DFS configuration file is being used.
- The **dfsbind** process is described in Section 3.1.7.

The following text describes the DFS configurations: a System Control machine, a FLDB machine, a File Server machine, a Binary Distribution machine, and a DFS client that is also a private file server machine.

A system control machine distributes system configuration information, such as administrative lists, which is shared by all DFS server machines in an administrative domain. This machine runs the **upserver** process and the **bosserver** process.

A FLDB machine runs the **flserver** process. The FLDB machine tracks the locations of all filesets and records the locations of filesets in the FLDB. The **flserver** process can run on the same machine as the File Server machine.

A File Server machine is used to export DCE LFS and non-LFS data for use in the global namespace. This machine must run the **fxd**, **ftserver**, **bosserver**, and **repserver** processes. File Server machines also run the **upclient** process to receive configuration file updates. The client process, **dfsbind**, must also run on this machine. The full range of fileset operations, including replication, is available on this machine.

Similarly, the Binary Distribution machine stores and distributes DFS binaries for processes and command suites to all other server machines of its Central Processing Unit (CPU) or Operating System (OS) type.

As previously explained, a DFS client machine runs the **dfsd** and **dfsbind** processes. Optionally, a DFS client machine can be configured as a private File Server in order

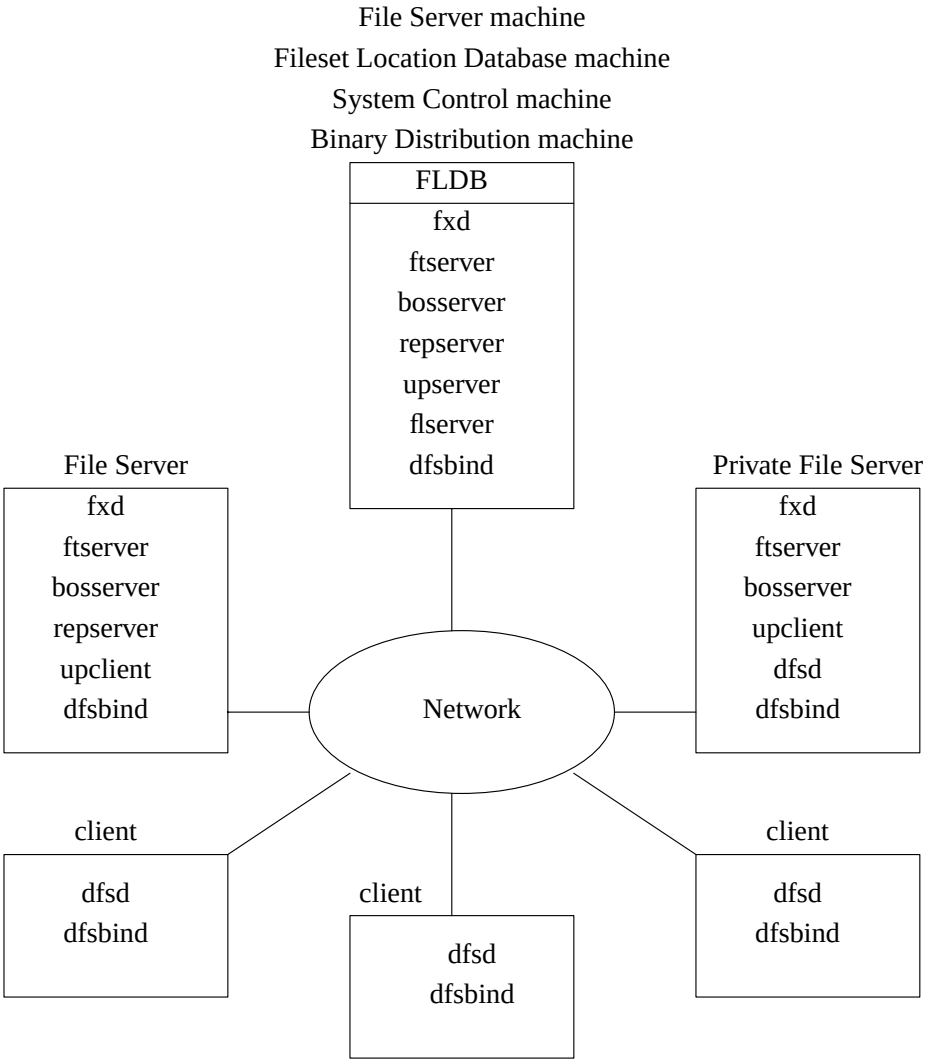
to export its local file system for use in the global namespace. This machine must run the **fxd** and **ftserver** processes. It is recommended that you also run the **bosserver** process.

A private File Server machine is controlled by the owner of the machine, not by the system administrator. The purpose of a private File Server machine is to allow individual users to export a small number of filesets. For further information about this configuration, see the discussion of DFS client machines in the *OSF DCE DFS Administration Guide and Reference*.

Figure 3-1 shows a DFS configuration that uses a File Server machine to run the FLDB machine, a System Control machine, and a Binary distribution machine. A second machine is a file server machine only. One DFS client machine is configured as a private file server in order to export filesets for use in the global namespace. Note that the first machine is configured to perform multiple roles.

Note: Figure 3-1 shows DFS alone. In addition, each client would run the processes previously described in this chapter. A complete cell would also include servers for the minimum DCE configuration.

Figure 3–1. An Example DFS Configuration



For information about other DFS configuration options, see the *OSF DCE DFS Administration Guide and Reference*. The *OSF DCE DFS Administration Guide and Reference* describes an additional DFS server role, the backup database machine.

A backup database machine stores the backup database and other administrative information used in the DFS Backup System.

3.3 DCE Administration Utilities

The following subsections describe the utility programs that DCE provides for managing and maintaining the DCE software. The last subsection tells you which utilities to place on a machine that is specially configured for the remote administration of DCE servers.

3.3.1 DCE Control Program

The overall administration tool for DCE, **dcecp**, has functions for administering the DCE services. You cannot use the program to administer GDS and DFS.

The **dcecp** utility is included in all of the DCE server software packages, except GDS and DFS.

3.3.2 Security Service Administration Programs

The Security Service provides the following administration utilities:

- The **sec_create_db** utility creates the Security database and sets up some configuration files.
- The **sec_salvage_db** command helps you recover from possible program errors and data corruption. The **sec_salvage_db** command applies internal consistency checks to the Security database (registry) and fixes internal data structure problems. You can also use **sec_salvage_db** to generate an ASCII version of the registry that can be edited and reconstructed, if necessary.
- The **passwd_import** command allows you create registry entries based on the group and password files from machines that do not implement DCE Security.
- The **passwd_export** command allows you to update the UNIX **/etc/passwd** and **/etc/group** files with current user information obtained from the registry.

- The **passwd_override** file allows you to establish overrides to the information contained in the registry.

3.3.3 CDS Administration Programs

CDS provides the CDS Browser (**cdsbrowser**) utility, which is a Motif-based program useful for viewing the contents and structure of the CDS namespace.

3.3.4 GDS Administration Programs

The **gdsditadm** program supports administration of the contents of the local DUA cache database and the GDS database, both local and remote.

3.3.5 DFS Administration Programs

DFS provides the following administration utilities:

- The **salvage** process checks the DCE LFS file system for internal consistency and corrects errors it finds.
- The **fts** commands help you manage filesets.
- The **bak** commands help you perform backup tasks.
- The **cm** commands help you customize the performance of the Cache Manager and examine features of DFS.
- The **bos** commands help you contact the Basic OverSeer (BOS) server used to monitor processes on server machines in your cell. You can also use the **bos** commands to perform some Security tasks.
- The **scout** program helps you monitor the File Exporters running on File Server machines. You may want to install **scout** only on the system administrator's DFS client machine.
- The **newaggr** command can format a raw disk partition for use as a DCE LFS aggregate.

- The **dfsexport** command makes DCE LFS aggregates and non-LFS partitions available to remote users through use of the File Exporter.

3.3.6 Programs for DCE Remote Administration Machines

You may decide to configure a machine for the remote administration of the DCE servers. The administration utilities that need to be installed, in addition to the DCE client software, are the following:

- The **passwd_override**, **sec_admin**, **sec_create_db**, and **sec_salvage_db** programs must be installed for Security service administration.
- The **gdsditadm** and **gdscp** for GDS administration.
- The **bak**, **bos**, **cm**, **dfsexport**, **fts**, **newaggr**, **salvage**, **scout** programs for DFS administration.

No software other than **dcecp** and the DCE client software is needed for administering DCE RPC, DTS, Security, or CDS.

3.4 Application Development Environment Machine

A DCE machine can also be configured for the development of DCE applications. This involves adding to the basic DCE client configuration several include (**.h**) and interface specification (**.idl**) files, along with the **idl** program.

Chapter 4

Location of Installed DCE Files

This chapter describes the location of DCE files that are created during the installation and configuration processes. The files used by DCE are grouped in the following locations:

- The *dceshared* subdirectories
- The *dcelocal* subdirectories
- Conventional UNIX subdirectories

Some information needs to be kept locally on a machine for reliability and to ensure that security is maintained. For example, when you configure DCE, the file that contains the name of your cell must be on the machine that is being configured. This file is stored in the *dcelocal* subtree. Other information used in DCE can be and needs to be shared among machines in a cell. For convenience, this information is stored in the *dceshared* subtree.

The *dceshared* subtree is created when you use the **tar** command to retrieve the archived files from the DCE tape, as described in the *OSF DCE Release Notes*. The *dcelocal* subtree is created when you install DCE components, as described in Part 2.

The complete set of delivered DCE files, except those that are created during run time, are stored under *dceshared*. The *dcelocal* subtree is a subset of *dceshared*. Files that are required in conventional UNIX subdirectories and executables that are required in *dcelocal* subdirectories can be duplicates of files and templates in the original *dceshared* subtree. In some cases, files are installed into directories such as **/usr/lib**, **/usr/bin**, or **/bin** for performance reasons. In other cases, symbolic links can be used from the conventional UNIX subdirectories to *dcelocal*.

The following sections describe the DCE subdirectories. Appendix B provides the directory layout for *dceshared*, *dcelocal*, and the conventional UNIX subdirectories that are used by DCE.

4.1 The dceshared Subtree

The files in the *dceshared* subtree can be kept on local machines or, preferably, they can be exported to other machines in the DCE cell by using DFS. Therefore, sharable files, including binaries that are addressed by **@sys**, are stored under *dceshared*. The *dceshared* subtree is read-only.

All files generated by a DCE build, all files delivered to binary licensees, and if appropriate, all files delivered to source licensees are initially stored in the *dceshared* subdirectories. All files in the *dceshared* subdirectories are kept unmodified over the lifetime of an installed DCE release. Configuration and data files are only stored as templates in *dceshared*. The actual working set of data files is located in the *dcelocal*/**var** and *dcelocal*/**etc** subdirectories.

The default pathname prefix for *dceshared* is **/opt/dce**, which is a symbolic link to **/opt/dce1.0**, or for DFS, to **/:/opt/dce1.0** (the short form of **/.:/fs/opt/dce1.0**). This entry is always physically located at the local machine. Therefore, the local system administrator (or the respective software administrator) must have write permission to modify this link. You can redirect this link from the fileset on the local machine to the cell-wide accessible fileset as soon as the local machine is configured and the cell is available; for example, redirect **/opt/dce** to **/:/opt/dce1.0**.

Note: Special care must be taken because this link is crucial for protecting the local machine if it is running as client and for protecting the server machine if it is acting as service provider. This symbolic link must be created in a protected

directory, which is comparable to **/etc**. Only the local system administrator needs to have write and modify permissions to this directory.

To avoid having replicas of *dceshared* files on local machines, you can use a symbolic link to access the cell-wide versions of these files. In case DFS users do not want to have replicas of these files physically stored on their local machine, they can remove the *dceshared* subtree that is installed on the local machine and redirect the default symbolic link to the cell-wide *dceshared* subtree, if these particular files are available there. The subdirectory that *dceshared* points to has a version number associated with the pathname that provides the capability of running multiple versions of DCE in one cell. This capability is sometimes required in an intermediate phase of upgrading to a new release. An additional advantage is a simplified deinstallation procedure.

If necessary, you can create copies or symbolic links from the other locations to **/opt/dce/***, such as *dcelocal*/**var**, *dcelocal*/**etc**, *dcelocal*/**bin**, and **/usr/bin**. These guidelines are based on the assumption that you want to use the DCE capability of cell-wide file sharing. The pathname for *dceshared* is set at compile time and is not associated with any particular version number.

4.2 The dcelocal Subtree

In order to initially boot a server and configure the cell, the appropriate files for mandatory servers (CDS and Security) need to be available on that server machine (in the *dcelocal* subtree). It is strongly recommended that copies of the minimum set of programs and data files that were installed during the default DCE installation procedure be kept locally on server machines for standalone operation and emergency maintenance.

The contents of the *dcelocal* subtree can vary from machine to machine inside a DCE cell to accommodate and serve specific configurations. In addition, every machine must have local access to certain files so that each machine can run as a standalone system if the machine is disconnected or partitioned from the cell. The appropriate files on DCE servers that have to be local to the server machine must be stored under *dcelocal*. Client-related data files are stored under *dcelocal*/**etc** (static configuration data) and *dcelocal*/**var/adm** (log files and so forth). All server-specific data files are located in the *dcelocal*/**var**/*dce-component-name* directory.

The default pathname for *dcelocal* is set to **/opt/dcelocal** during the configuration process. This is a fixed pathname. Every machine must have local access to the files that are necessary to configure it, up to activating DFS access in the cell. The **/opt/dcelocal/dce_cf.db** file is the DCE configuration file containing the name of the host to be configured and the cell name. A machine must access this small set of DCE files, which is kept on the machine's local disk, to start up the various DCE components and for local configuration information and log information.

Because DCE configuration takes place after mounting the local file systems, none of these files has to be available in the root partition.

During DCE configuration, only the executables in *dcelocal*/**bin** are reliably available. Start-up procedures, such as **rc** scripts, need to address executables through *dcelocal*/**bin** rather than **/usr/bin**, even if the same files are believed to be in both directories. Commands in **/usr/bin** can be just symbolic links to *dcelocal*.

The *dcelocal* subtree is populated and initialized during DCE configuration.

4.3 Conventional UNIX Directories

Some files and directories used by DCE are accessible in conventional UNIX directories. These DCE files and directories need to be accessible in conventional locations so users can conveniently access frequently used utilities and data, such as **idl** from the **/usr/bin** directory and **localtime** from the **/etc/zoneinfo** directory. Header files are accessible in **/usr/include** or in its subdirectory, **/usr/include/dce**, and libraries, such as **libdce.a**, are kept in **/usr/lib**.

4.4 UNIX Permissions for DCE Subdirectories

All directories in the file system are created with UNIX permission set to **rwxr-xr-x** for user **root** and group **bin**. In subsequent configurations, the DCE Security Service can define roles for several administrators (principals or groups). A possible scenario follows:

- A software administrator who owns the installed software packages and has write and modify permissions for the entire set of files included in *dcelocal*.

- DCE service administrators who are responsible for a particular DCE service such as Security and have read and write permissions for the data files for the respective service. You can assign a separate DCE Security Service administrator, while a single cell administrator can have responsibility for the remaining DCE services.
- A local DCE system administrator who is responsible for client setups and has read and write permissions for the respective local files.

Chapter 5

Overview of DCE Maintenance

Once you have performed the tasks required for planning, installing, and configuring your DCE system, you can go on to perform the tasks required for maintaining the system. The initial tasks of planning, installing, and configuring are performed infrequently, some only once. Maintenance tasks, however, are performed on a regular basis throughout the lifetime of your system.

Maintenance of a distributed system includes the following areas:

- Performance tuning
- Configuration control
- Security and access control

This chapter summarizes some of the primary DCE administration tasks. The first section of this chapter tells you how to start up DCE. The remaining sections describe tasks that apply to the individual components of DCE. DCE component tasks are documented in detail in the *OSF DCE Administration Guide—Core Components*, *OSF DCE DFS Administration Guide and Reference*, and *OSF DCE GDS Administration Guide and Reference*.

5.1 Starting Up DCE

The **dce_config** script, which is described in Chapters 6, 7, and 8, creates an **/etc/rc.dce** file to start up the DCE processes. This file is customized to an individual machine's configuration. To start up the DCE processes, enter the following command:

sh /etc/rc.dce

5.2 Changing the Network Address of a DCE Machine

Occasionally, a machine running DCE will need to change its network address. DCE stores the network address in several files which need to be updated. To change a DCE machine's network address, you must perform the following procedure:

1. Shutdown DCE (by selecting option 4 in the **dce_config** Main Menu).
2. Change the machine's network address and reboot the operating system, as needed.
3. Remove the machine's CDS cache:

   rm /opt/dcelocal/var/adm/directory/cds/cds_cache*

4. Update the network address in the **/opt/dcelocal/etc/security/pe_site** file.
5. Update the network address in the **/opt/dcelocal/etc/cds_config** file.
6. Restart DCE (by selecting option 3 in the **dce_config** Main Menu).
7. If necessary, update the following CDS objects:

   /.:/hosts/*hostname*/**self**
   /.:/hosts/*hostname*/**dts-entity**

   These CDS objects contain bindings that are exported when the cell is configured by the **dce_config** script, but are not checked when daemons are restarted.

   In order to ensure that the **self** and **dts_entity** objects have the proper updated address, run the **dcecp object show** command for each object:

```
dcecp>  object show /.:/hosts/hostname/object
```

If the network address listed in the **Tower** attribute is incorrect, change it to an appropriate one.

To update the network address of the **self** object, use the **dcecp rpcentry unexport** command to unexport the object's binding:

```
dcecp> rpcentry unexport /.:/hosts/hostname/object -interface interface_id \
> object if_uuid
```

The UUID value *if_uuid* for the object can be obtained by running the following command:

```
dcecp> rpcentry show /.:/hosts/hostname/object -interface interface_id \
> object if_uuid
```

Now update the object with the correct (new) network address. The command to use is:

```
dcecp> rpcentry export /.:/hosts/hostname/object -interface interface_id \
> -binding protocol_sequence -object if_uuid
```

The procedure for updating the **dts-entity** object's network address differs from that for other CDS objects. First, stop the **dtsd** process on the target host. Then, delete the **/.:/hosts/***hostname***/dts-entity** object (by using the **dcecp object delete** command). Finally, you must restart the **dtsd** process (using **dce_config**).

8. If DFS is used in a DCE cell, you must also issue the following command for every DFS server (**flserver**) that has undergone a network address change:

   ```
   > fts edserver -server old.ip.addr -changeaddr new.ip.addr
   ```

   You can then restart all of the DFS servers in the cell so that they recognize the new address. You may also have to stop and restart DCE (using **dce_config**), as well, and remove the CDS cache before restarting the servers. For instructions on how to remove the CDS cache, see the *OSF DCE Administration Guide—Core Components*.

5.3 CDS Maintenance Tasks

CDS components, including clerks, servers, and clearinghouses, are largely self-regulating. Except for routine monitoring, CDS requires little intervention for system administration. When intervention is required, CDS provides system administration tools to help you monitor and manage the CDS namespace and CDS servers.

You can use the DCE control program (**dcecp**) commands described in Chapter 15 of the *OSF DCE Administration Guide—Core Components* to create and manage the components of a CDS namespace.

You can also manage CDS by using the CDS Browser utility (**cdsbrowser**) to view the namespace. The **cdsbrowser** utility enables you to monitor growth in the size and number of CDS directories in your namespace. You can use the **cdsbrowser** utility to display an overall directory structure, as well as the contents of directories. Chapter 19 of the *OSF DCE Administration Guide—Core Components* discusses the **cdsbrowser** utility.

If you have a large organization, you can improve efficiency by having one system administrator responsible for CDS servers and another system administrator responsible for the namespace. You can delegate responsibility for a subtree of the namespace to another administrator by granting access control rights to that person.

For more detailed information on CDS maintenance tasks, see the *OSF DCE Administration Guide—Core Components*.

5.3.1 Monitoring CDS

CDS monitoring tasks fall into the following two categories:

- Monitoring the namespace
  - — Monitor the size and usage of clearinghouses and determine the need for new CDS servers and clearinghouses. Plan and oversee the configuration of these new servers and clearinghouses.
  - — Maintain and monitor a map of the namespace.
- Monitoring CDS servers

— Enable event logging, monitor CDS events, and solve system-specific problems if they arise. If necessary, notify the namespace administrator of problems that can affect other CDS servers or clerks.

— Monitor the success of skulks that originate at the server. A *skulk* is a method of updating all replicas through repeated operations.

— Monitor the size and usage of the server's clearinghouse and, if necessary, discuss with the namespace administrator the need to relocate some replicas or create a new clearinghouse.

— Monitor and tune system parameters that affect or are affected by CDS server operation.

Note: When monitoring memory usage for CDS servers, it is important to understand that memory remains allocated under certain conditions. Memory associated with objects remains allocated until a skulk is successfully completed. Memory associated with directories remains allocated until the server is disabled and restarted.

For detailed discussions of these tasks, see Chapter 17 of the *OSF DCE Administration Guide—Core Components*.

5.3.2 Managing CDS

CDS management tasks fall into the following two categories:

- Managing the namespace

  — Oversee the creation of new directories and assign names according to a standard, or enforce established guidelines in the assigning of names. (Beyond a certain level in the directory hierarchy, you can delegate the responsibility of creating and maintaining directories. You need to keep track of the new directories being created to make sure they are appropriately replicated.)

  — Determine the default access control policy.

  — Administer and enforce the established access control policy for directories and entries.

  — Determine where and when new replicas of a directory are necessary.

  - — Create soft links for objects that change locations or for objects that need to be renamed. An *object* is a resource, such as a disk, an application, or a node, that is given a CDS name. A name plus its attributes make up an *object entry*. A *soft link* is a pointer that provides an alternate name for an object entry.

    Publicize and encourage the use of the new names so that eventually the soft links can be deleted.
  - — Solve or direct the resolution of problems involving multiple CDS servers.
- Managing CDS servers
  - — Manage access control on directories and objects, and monitor the size and usage of directories in the server's clearinghouse. Create new directories, possibly with the namespace administrator, when necessary.
  - — Create new objects in directories or oversee their creation. (Beyond a certain level in the directory hierarchy, you also can delegate the responsibility of maintaining directories and the objects in them.)
  - — Add new administrators to the **cds-admin** security group.

Chapters 16, 17, 18, and 20 of the *OSF DCE Administration Guide—Core Components* provide detailed information about how to perform these tasks.

5.3.3 CDS Security and Access Control

The DCE control program (**dcecp**) and CDS ACL Manager work together to manage authorization in CDS. To modify, add, delete, or view ACL entries in the CDS namespace, use **dcecp acl** commands. When **dcecp** issues a request to perform an operation on a CDS object, the CDS ACL Manager checks permissions, based on ACL entries, and grants or denies the request. The CDS ACL Manager is an integral part of the **cdsd** and **cdsadv** processes.

Chapter 16 of the *OSF DCE Administration Guide—Core Components* provides detailed information about handling CDS security and access control, including guidelines for setting up access control in a new namespace.

5.4 GDS Maintenance Tasks

GDS provides a menu-driven interface for performing the maintenance tasks described in the following subsections. For more detailed information on the GDS maintenance tasks, see the *OSF DCE GDS Administration Guide and Reference*.

5.4.1 Monitoring GDS

You can monitor GDS by displaying directory system status information with the GDS interface or by using the trace system to log directory processes.

The GDS interface allows you to do the following:

- Display directory system status information, which shows if a directory is active or inactive, what processes are available, how many processes are available, and if the trace system is active or inactive.
- Activate the trace system, which starts the trace system for logging directory processes.
- Deactivate (or stop) the trace system.

For more information on these functions, see the *OSF DCE GDS Administration Guide and Reference*.

GDS maintains log files for each of the following processes:

- The DUA process
- The Cache process
- The Client-stub process
- The Server-stub processes
- The DSA processes
- The GDS system administration process
- The Monitoring process

For more information on the location, contents, and creation of log files, see the *OSF DCE GDS Administration Guide and Reference*.

5.4.2 Managing GDS

GDS provides the following functions for management tasks:

- Administrative Functions

  This directory management function, which is available after logging into a DSA, supports administration of the contents of the GDS database, both local and remote. You can choose from the four types of administration functions in the following list. These administration functions allow you to administer objects, shadows, trees, and schema for your directory service system. A GDS *schema* is a set of rules and constraints for tree structure, object class definitions, attribute types, and syntaxes that characterize the Directory Information Tree (DIT).

  — Object Administration

    This function controls objects and changes their attributes.

  — Schema Administration

    This function modifies the Object Class Table, Structure Rule Table, and Attribute Table to store a new schema in the Directory System Agent.

  — Shadow Administration

    This function executes, schedules, or modifies shadow jobs for updating.

  — Tree Administration

    This function adds, deletes, or modifies subtrees.

- Administrative Functions Under the DUA Cache

  This directory management function supports the administration of the contents of the local DUA cache database. You can choose from the two types of administration functions in the following list. These administration functions allow you to administer the objects stored in the local DUA cache database and the cache update job.

  — Object Administration

    This function controls objects stored in the local DUA cache database and changes their attributes.

— Cache Update

This function displays, activates, or deactivates the Cache Update job, or changes its update frequency.

- Activation of a directory system installation

  This directory management function activates the directory installation by starting the background processes of GDS.

- Deactivation of a directory system installation

  This directory management function ends the background processes of GDS. All running directory processes are ended, but running operations are not interrupted, and the data consistency of the managed data is retained.

For more information on these directory management functions, see the *OSF DCE GDS Administration Guide and Reference*.

5.4.3 Backing Up GDS Data Files

You can back up directory system data files in GDS by selecting the entry that saves the local data files to diskette or tape from the function entries in the menu mask. The save process backs up all the local data files (local DSA data, DUA cache data) that belong to your directory system.

GDS has a password feature that protects the directory system data files on the data medium. The use of this password is optional when saving data, but you must use the password when loading files that are saved with a password.

For more information on saving directory system data files, see the *OSF DCE GDS Administration Guide and Reference*.

5.4.4 Changing Global Directory Configurations

The GDS interface has a function, `Configuration of a directory system`, that enables you to enter, delete, display, or change configuration data,

such as the number of clients or servers to be activated. For more information on this function, see the *OSF DCE GDS Administration Guide and Reference*.

5.5 DTS Maintenance Tasks

Like CDS, DTS is largely self-regulating once configuration of the service is complete. However, there are times when you need to intervene. Use **dcecp** to perform the following DTS configuration and management tasks:

- Identify system clock problems.
- Adjust the system clocks.
- Change DTS attributes for varying WAN conditions.
- Modify the system configuration when the network environment changes.

For more detailed information on DTS maintenance tasks, see the *OSF DCE Administration Guide—Core Components*.

5.5.1 Managing the Distributed Time Service

You can use **dcecp** to create and enable DTS. Once this is done, you can perform routine management tasks, such as enhancing performance, reconfiguring the network, and changing local time.

Several commands and characteristics modify and improve the performance of your network. The **dts modify** command changes the values of many of these characteristics. The **dts show** command displays the values of characteristics at any time. The following are some of the tasks you can accomplish using the DTS commands and the characteristics of DTS that can be set:

- Display or change the number of servers that must supply time values to the system before DTS can synchronize the system clock.
- Display or change the inaccuracy limit that forces the system to synchronize in order to bring the inaccuracy back to an acceptable level.
- Display or change the interval at which you want clock synchronization to occur.

- Display or change the reaction to a faulty system clock.
- Display or change the settings that indicate how often to query servers.

Refer to the *OSF DCE Administration Guide—Core Components* for more information on these and the following tasks:

- Creating and enabling DTS.
- Assigning the courier role to servers to facilitate communications to other parts of your network.
- Matching the epoch number for servers that you add to your network after the initial configuration. An *epoch number* is an identifier that a server appends to the time values it sends to other servers. Servers only use time values from other servers with whom they share epoch numbers.
- Advertising DTS servers to CDS, thereby registering them as objects in the namespace.

5.5.2 Modifying System Time

Sometimes you need to modify the system time. You can update time to match the international time standard, Coordinated Universal Time (UTC), from a source such as telephone, radio, satellite, or another external referencer, if your network does not use time providers and the network systems have been running for some time. The **clock set** command accomplishes this task by gradually modifying the time.

The **clock set** command used with the **-abruptly** option and the **dts synchronize** command provide additional methods for adjusting the system clock and synchronizing systems.

5.6 Security Service Maintenance Tasks

The following subsections summarize the maintenance tasks you perform while administering the Security Service. For more detailed information on Security maintenance tasks, see the *OSF DCE Administration Guide—Core Components*.

5.6.1 Managing the Security Service

The Security Service management tasks include the following:

- Creating and maintaining accounts by using **dcecp**

  dcecp provides commands for creating and maintaining registry information, including persons, groups of users, and accounts.

  Keep the following things in mind when administering DCE accounts:

  — If you share files with other systems that do not use the registry, be sure that names, UNIX IDs, and account information are consistent between the registry and the foreign password and group files. Use **passwd_import** to identify and resolve any conflicts that exist. Chapter 39 of the *OSF DCE Administration Guide—Core Components* describes how **passwd_import** works.

  — If you maintain **/etc/passwd** and **/etc/group** files in standard UNIX format, you need to run **passwd_export** to make password, group, and organization files on local machines consistent with the registry. See Chapter 36 of the *OSF DCE Administration Guide—Core Components* for more information about the **passwd_export** command.

  — For principals in other cells to access objects in your cell, you need to set up a special account for the foreign cell in your cell's registry. This account indicates that you trust the Authentication Service in the foreign cell to correctly authenticate its users. Use the **dcecp registry connect** command to create an account for a foreign cell.

- Using ACLs

  Use the **dcecp acl** commands to display, add, modify, and delete ACL entries for a specific object in the cell namespace. (See the *OSF DCE Administration Guide—Core Components* for detailed information on how to use the **dcecp acl** command.)

- Setting and maintaining registry policies

  Registry policies include certain password and account information. Policies also include overrides, which are exceptions tied to a specific machine. Use the **dcecp registry** commands to set and maintain registry policies. Details on how

to these commands are in Chapter 35 of the *OSF DCE Administration Guide—Core Components*.

Ticket expiration date, password life span, password format, and password expiration date are examples of registry policies that you can set. If both an organizational policy and a registry policy exist for password format, for example, the more restrictive policy applies.

You can establish overrides to the information contained in the registry. Override information is stored in the **passwd_override** and **group_override** files on a local machine. The **passwd_override** file contains the home directory, the login shell, entries for overriding the password, and GECOS information, which is general information that is used by users but not required by the system, such as office and phone numbers. For details about how to edit the **passwd_override** file, refer to Chapter 36 of the *OSF DCE Administration Guide—Core Components*.

- Backing up the registry

  Chapter 36 of the *OSF DCE Administration Guide—Core Components* describes the back-up procedure to follow for the master registry site. When you restore the database, it is automatically propagated to the slaves.

- Setting up and maintaining Audit Service data

  Audit Service data includes event numbers, event class numbers, event class files, audit filters, and audit trail files. Use the **dcecp aud**, **audevents**, **audfilter**, and **audtrail** commands to manage Audit Service data. The *OSF DCE Command Reference* provides descriptions of audit-related **dcecp** objects and commands. See Chapters 42 and 43 in the *OSF DCE Administration Guide—Core Components* for more information about Audit Service administration.

- Troubleshooting

  When you encounter problems that cannot be resolved through routine management procedures, or when hardware failures stop the registry from operating, there are several troubleshooting procedures you can use. Chapter 40 of the *OSF DCE Administration Guide—Core Components* describes the following tasks:

  — Recreating a registry replica

  — Recovering the master registry

— Forcibly deleting a replica

— Adopting registry objects that are orphaned because their owner has been deleted

5.6.2 Reconfiguring the Registry

There are two main reconfiguration tasks included in the administration of the Security Service. The following tasks are described in Chapter 37 of the *OSF DCE Administration Guide—Core Components*:

- Changing the master registry site when you plan to move the machine that runs the master registry server from your network or shut the machine down for an extended period
- Removing a server host from the network when you plan to remove a machine that runs a slave registry server from the network or shut that machine down for an extended period

5.7 DFS Maintenance Tasks

The following subsections summarize the five major DFS maintenance tasks: monitoring DFS servers and clients, managing filesets in a cell, backing up filesets, reconfiguring the Cache Manager, and managing DFS security. For more detailed information on DFS maintenance tasks, see the *OSF DCE DFS Administration Guide and Reference*.

5.7.1 Monitoring DFS Servers and Clients

You can monitor DFS in the following ways:

- Use the BOS server to continually monitor DFS server processes. You define which processes the BOS server monitors, and you control server process status by issuing **bos** commands to perform routine maintenance or to correct errors in addition to those the BOS server handles. You can also use **bos** commands to

restart DFS weekly. See the *OSF DCE DFS Administration Guide and Reference* for details about the BOS server and how to use **bos** commands.

- Use the **scout** program to monitor the File Exporter, which runs on File server machines. The File Exporter makes DFS files available to client machines. The **scout** program collects and displays information about the machines you select to monitor. It displays information such as disk usage and the number of connections a machine has.

5.7.2 Managing Filesets in a Cell

The basic unit of administration in DFS is the fileset, which is a collection of related files. The *OSF DCE DFS Administration Guide and Reference* describes the following tasks:

- Creating read/write filesets
- Replicating filesets
- Creating backup filesets
- Mounting and naming filesets
- Listing information about filesets
- Moving filesets
- Examining the FLDB entry
- Salvaging filesets
- Synchronizing fileset information
- Setting and listing fileset quota and current size
- Removing filesets and their mount points
- Dumping and restoring filesets
- Renaming filesets
- Unlocking and locking FLDB entries

5.7.3 Backing Up Filesets

The system administrator uses the Backup System provided by DFS to make backup tape copies of filesets. For a discussion of how often to perform backups using **bak** commands, which filesets need to be backed up, and when to make full or incremental backups, see the *OSF DCE DFS Administration Guide and Reference*. The *OSF DCE DFS Administration Guide and Reference* also describes the following tasks:

- Configuring a backup machine
- Installing tape coordinators
- Listing information from the backup database
- Creating and reading tape labels
- Performing a backup
- Performing a restore
- Canceling backup and restore operations

5.7.4 Reconfiguring the Cache Manager

Usually, all Cache Manager machines are configured in the same way, but you can change certain features to achieve different levels of performance across client machines. You can use the **cm** commands to perform the following tasks:

- Directing the Cache Manager to use machine memory instead of disk space for caching
- Changing the cache size
- Changing the cache location
- Altering the default size and numbers of chunks that compose a cache
- Directing the Cache Manager to allow programs that reside in foreign cells to execute with **setuid** status
- Changing the cell membership
- Forcing the Cache Manager to discard or fetch a new version of a file or directory from the File server machine

5.7.5 DFS Security and Access Control

In DFS, you can set up administrator groups with special privileges that permit members of a group to do the following:

- Issue administrator commands.
- Create or remove filesets.
- Perform system backups.

In DFS, administrative lists define the principals that can perform actions affecting specific server machines. Use the **bos** commands to create and maintain administrative lists. Use **dcecp** to create administrative groups and place these groups on administrative lists. Adding and removing users from groups rather than altering the administrative lists themselves simplifies system administration.

Groups of DFS server machines that are administered as a single unit are known as DFS administrative domains. Whenever you add or remove server machines in a DFS domain, you must also alter the keytab file for that machine. A keytab file contains a server encryption key, which is used to provide security between servers and their clients. Use the **bos** commands to maintain a server's keytab file.

To verify or modify ACLs, use the **dcecp acl** commands.

5.8 Shutting Down DCE

When you wish to shut down the DCE processes, you can use the **dce_config** script (option 4 in the Main Menu), which calls the **dce_shutdown** script, or you can run the **dce_shutdown** script directly. To run the **dce_shutdown** script, enter the following command:

```
sh /etc/dce_shutdown
```

Part 2

Configuring and Starting Up DCE

Chapter 6

Overview of the dce_config Script

The **dce_config** script (and the component scripts it invokes) are a tool for installing and configuring DCE machines. Using the menu-driven **dce_config** script you can perform the following tasks:

- Install the DCE binaries on client and server machines
- Configure and start DCE servers
- Restart DCE servers
- Stop DCE servers
- Unconfigure DCE client machines by removing hosts from the CDS and Security databases
- Remove the data files created by DCE servers

Note: The **dce_config** script is supplied by OSF as part of the DCE offering. Your system vendor may provide an alternative method of installing the DCE software. If so, refer to the vendor's documentation for instructions on installing DCE software.

This chapter is an overview of how to use **dce_config**. See Chapters 7 and 8 for instructions on installing DCE binaries and configuring clients and servers. See Chapter 9 for instructions on restarting, stopping, unconfiguring, and removing servers, and see Chapter 10 for information on automating **dce_config** processing, **dce_config** environment variables, message logging control, and the **dce_config** component scripts.

This chapter describes:

- How to start **dce_config**
- How **dce_config** displays defaults
- Messages displayed and logged by **dce_config**
- How to exit from the **dce_config** script

6.1 Starting the dce_config Script

To start the **dce_config** script, perform the following steps:

1. Login as root to the machine on which you are installing or configuring DCE. You cannot install or configure machines remotely.
2. If necessary, copy the **/etc** directory from the distribution media by performing the following steps:
   a. Use the **cd** command to move to the **/opt/dce** directory.

      cd /opt/dce

   b. Use the **tar** command to copy the **dce1.1/etc** directory from the tape.

      tar -xvf *media device* **dce1.1/etc**

3. Invoke **dce_config** by typing:

   dce_config -i

   and pressing <**Return**>.

The **-i** option tells **dce_config** to look in the current directory (which should be **/opt/dce1.1/etc**) for the component scripts it needs to run. After you have invoked **dce_config** once with the **-i** option, you do not need to use the option again.

Note: On machines running OSF/1 Release 1.1.1 only, you must invoke **dce_config** as an argument to **ksh** for the return function to work correctly. To do this, type **ksh dce_config** and press <**Return**>. This is not necessary for OSF/1 Release 1.2.

4. The **dce_config** script displays the DCE Main Menu, which lists all the functions you can perform with **dce_config**.

```
DCE Main Menu ( on host_name )

 1. INSTALL     -install dce software
 2. CONFIGURE   -configure and start DCE daemons
 3. START       -re-start DCE daemons
 4. STOP        -stop DCE daemons
 5. UNCONFIGURE -remove a host from CDS and SEC databases
 6. REMOVE      -stop DCE daemons and remove data files
                 created by DCE daemons

99. EXIT

selection:
```

To choose a function from the DCE Main Menu and from any of the **dce_config** menus, type its associated number at the `selection:` prompt and press <**Return**>.

All **dce_config** menus display the name of the node on which you are running **dce_config**. In the sample menus shown in this guide, the actual node name is represented by *host_name*.

6.2 Defaults

The **dce_config** script prompts you for information it needs. You supply that information by typing it in after the prompt and pressing **<Return>**. When **dce_config** prompts you for information, it shows the default value in parentheses just after the prompt. For example, in the following prompt for the location of the message catalogs, the default is **/usr/lib/nls/C**.

```
Enter the directory into which message catalogs
should be stored on the local machine
(/usr/lib/nls/C):
```

To accept a default value, press **<Return>** without typing in any other information.

6.3 Messages and Message Logging

As the **dce_config** script processes, it displays messages that inform you of actions being taken and errors encountered. You can receive the following six types of messages: error, warning, summary, detail, verbose, and debug.

The messages for error, warning, and summary errors are displayed on your screen by default. You can designate which type of messages are to be displayed, though, by setting environment variables (See Chapter 10.) All messages are also recorded in the **dce_config** log file. The following subsections describe each type of message that the **dce_config** script generates.

6.3.1 Error Messages

Error messages inform you that an unexpected, possibly fatal, error has occurred. When you receive an error message, you must either press **<Return>** to continue processing or **<CTRL-C>** to exit. Error messages have the form:

```
ERROR:  message text
        Press <RETURN> to continue, CTRL-C to exit:
```

A sample error message follows:

```
ERROR:   Can't create file /opt/dcelocal/ext/dfs_episode.ext
         Press <RETURN> to continue, CTRL-C to exit:
```

6.3.2 Warning Messages

Warning messages inform you of non-fatal events that you should be aware of before you continue. When you receive a warning message, you may be required to either: 1) press **<Return>** to continue processing or 2) **<CTRL-C>** to exit **dce_config**. The actual course of action taken by **dce_config** after it displays a warning message is determined by the setting of the **DO_CHECKS** environment variable. As supplied by OSF, the variable is set to prompt you to press **CTRL-C** or **<Return>**.

Warning messages have the form:

```
WARNING:  message text
```

The following output contains sample warning messages:

```
S:****** Attempting to stop all running DCE daemons...
You do not have any network credentials. All requests will be unauthenticated.
You do not have any network credentials. All requests will be unauthenticated.
WARNING: cdsadv not killed
WARNING: cdsd not killed
WARNING: dced not killed
WARNING: cdsclerk not killed
ERROR:   Failed to stop all running DCE daemons.
        Press <RETURN> to continue, CTRL-C to exit:
```

6.3.3 Summary Messages

Summary messages are a high-level summary of an action or the results of an action taken by **dce_config**. Summary messages have the form:

```
S:******  message text
```

A sample summary message follows:

```
S:****** Waiting for node self identity to be established.
```

6.3.4 Detail Messages

Detail messages show all actual commands that affect the configuration or the state of the machine being configured. The messages also show which **dce_config** component script executed the command. Detail messages that contain the word "Executing" provide a record of the exact commands used to configure a machine. Detail messages have the form:

```
D: message text
```

A sample detail message follows:

```
D:   dfs.clean: Executing: /opt/dcelocal/bin/bos stop -server
     /.:/hosts/antoine -process bakserver -wait >/dev/null 2>&1
```

6.3.5 Verbose Messages

Verbose messages are a summary of actions being taken by **dce_config** and the user. The messages show **dce_config** prompts and user responses, and all actual commands executed by **dce_config** and the subcomponent script that executed them. They provide a complete record of the user entries. Verbose messages have the form:

```
V:  message text
```

Some sample verbose messages follow:

```
V:   dfs.rm: Removing files created by DFS daemons on initial
     configuration.

V:   User query: Do you wish to first remove all remnants of previous DCE
```

```
    configurations for all components (y/n)? You should do so only if
    you plan on re-configuring all existing DCE components now: (n)

V:  User entry: y
```

6.3.6 Debug Messages

Debug messages show actual commands executed by **dce_config** and the component scripts that committed them. If a command does not execute successfully, the command's error message text is passed for display to **dce_config**. Do not confuse this error text with **dce_config** errors. Only **dce_config** error and warning messages indicate a **dce_config** error.

Debug messages have the form:

```
DEBUG:   message text
```

Some sample debug messages follow:

```
DEBUG:   Executing: daemon_slayer(dtsd)
DEBUG:   dtsd can't be killed, not running
```

6.3.7 The dce_config log File

In addition to being displayed on the screen, messages are also written to the **dce_config** log file, **/tmp/dce_config.log**. As with the screen displays, you can control the type of messages logged in the file using the environment variables described in Chapter 10. By default the log file is named **dce_config.log** and resides in the **tmp** directory. You can specify a different name and location with the **LOGFILE** environment variable described in Chapter 10.

A portion of a sample log file is shown in Figure 6-1.

Note that the sample is in four basic sections:

- The first section displays the name of the machine being configured (indicated by *host_name*)

- The second section displays the **dce_config** version number (indicated by *version_num*) and time and date **dce_config** was run. (indicated by *date_and_time*)
- The third section shows the settings of the environment variables described in Chapter 10.
- The final section shows a portion of the actual messages produced during a machine configuration

```
dce_config logfile for host host_name
****************************************************************
dce_config V: version_num executed: date_and_time
****************************************************************
V:          EXIT_ON_ERROR:          n
V:          DO_CHECKS:              y
V:          REMOVE_PREV_INSTALL:    <not set>
V:          REMOVE_PREV_CONFIG:     <not set>
V:          SEC_SERVER:             <not set>
V:          SEC_SERVER_IP:          <not set>
V:          UNCONFIG_HOST_PRESET:   <not set>
V:          CELL_NAME:              <not set>
V:          KEYSEED:                <not set>
V:          CACHE_CDS_SERVER:       <not set>
V:          CACHE_CDS_SERVER_IP:    <not set>
V:          REP_CLEARINGHOUSE:      <not set>
V:          HOSTNAME_IP:            <not set>
V:          NTP_HOST:               <not set>
V:          MULTIPLE_LAN:           <not set>
V:          LAN_NAME:               <not set>
V:          CONFIG_DFS_CLIENT:      <not set>
V:          CELL_ADMIN:             NULL
V:          CELL_ADMIN_PW:          <not set>
V:          TOLERANCE_SEC:          120
V:          check_time:             y
V:          DEFAULT_MAX_ID:         32767
V:          UID_GAP:                100
V:          LOW_UID:                <not set>
V:          GID_GAP:                100
V:          LOW_GID:                <not set>
V:          SYNC_CLOCKS:            <not set>
```

```
V:         DEFAULT_PW:           <not set>
V:         FILESYSTEM:           <not set>
V:         MEDIA:                <not set>
V:         DTS_CONFIG:           <not set>
V:         CP_OR_SYMLINK:        <not set>
V:         USE_DEF_MSG_PATH:     <not set>
V:         User query: DCE Main Menu
V:         User entry: 2
V:         User query: DCE Configuration Menu (on host_name)
V:         User entry: 1
S:****** Configuring initial cell...
DEBUG:     Executing: settimezone()
V:         DCE timezone already set.  To change it, consult the Release Notes.
V:         User query: Initial Cell Configuration menu (on node_name)
V:         User entry: 1
S:****** Configuring initial Security Server...
DEBUG:     Executing: config_sec()
V:         User query: Do you wish to first remove all remnants of previous DCE
           configurations for all components (y/n)? You should do so only if
           you plan on re-configuring all existing DCE components now: (n)
V:         User entry: y
S:****** Attempting to stop all running DCE daemons...
```

Figure 6–1. Sample Log File

6.4 Exiting from dce_config

Generally you will exit from **dce_config** by typing **99** at the DCE Main Menu and pressing <**Return**>. (If you need to return to the DCE Main Menu from another **dce_config** menu, type **98** and press **Return** at the menu's `selection:` prompt until you are back at the main menu.) Most of the **dce_config** submenus have a **99** choice for exiting the script.

You can exit from **dce_config** at any time and from any location by pressing **CTRL-C**.

Chapter 7

Installing DCE

This section is an overview of how to use the DCE installation and configuration script, **dce_config**, to install the DCE binaries built for your platform. Once the binaries are installed, you can configure the system as described in Chapter 8.

The **dce_config** script is supplied by OSF as part of the the DCE offering. Your system vendor may provide an alternative method of installing the DCE binaries. If so, refer to the vendor's documentation for instructions.

7.1 Prerequisites

Before you can install DCE on the local machine, you must be able to access the install tree containing the DCE binaries. Building the binaries is described in the *OSF DCE Release Notes*.

Additionally, you should read and understand the overview of DCE presented in Chapters 1 through 5 of this guide. Pay particular attention to Chapters 2 and 3, which can be used as a guide to planning cell configurations. These chapters describe

the space requirements for machines running DCE, the servers and client daemons required to be installed for each DCE component, the optional servers, tools, and utilities that make up each component, and other cell planning considerations. Note that because this chapter assumes you have read and understood these chapters, it makes no attempt to describe the function or use of DCE component software.

7.1.1 The Install Tree Location

You can install the DCE binaries from an install tree stored on your file system or from an install tree stored on a media device. If you install from a media device, **dce_config** must copy each file it needs from the device as it needs it. Because it is faster to install from the file system, we recommend that you use the **tar** command to copy the install tree to a directory named **/opt/dce** in your file system.

If you are installing from an install tree stored on the file system, ensure you have access to the tree either on the local machine or through a remote mount.

During the installation, you will be prompted to specify the location of the install tree. If it is on the file system, you supply its pathname. If it is on a media device, you supply the device name.

7.1.2 Machine Requirements

You must install the DCE binaries on each machine on which they will run. Ensure that each machine meets the following requirements:

- **Disk Space**—The machine must meet the disk space requirements outlined in Chapter 4 of this guide.
- **Required Ethernet Addresses**— Each machine must have an Ethernet address (a 12-digit hexadecimal number registered with the IEEE), usually contained on the Ethernet card. Some operating systems, like the AIX for example, provide routines to access the ethernet address from the card. Others, like OSF/1 for example, do not. For these machines, store the ethernet address in a file named **/etc/ieee_802_addr**. The file should contain the only the 12-hexadecimal digits address in text form, for example, `08002BFFFFFF`.

- **Wide Area Network (WAN) Connections**—If you will be configuring DCE over a WAN, set up the following ports for receiving packets on both ends of the WAN connection:
  - — **udp** port 88 for Kerberos
  - — **udp** port 135 for the **dced** daemon
  - — **tcp** port 135 for the **dced** daemon
  - — All **udp** and **tcp** ports greater than 1024 for all DCE services and applications

7.2 Installing DCE

During the installation, **dce_config** performs the following steps,

1. Installs the client and server executables in the correct directories.
2. Installs the message catalogs.
3. Creates administrative directories.
4. Installs the DCE library, **libdce**.
5. Defines the following variable names:
   - **DCEROOT** as **/opt**
   - **DCELOCAL** as **/opt/dcelocal**
   - **DCESHARED** as **/opt/dce**
   - **SUBSYSDIR** as **/subsys/dce**
   - **SECURITYDIR** as **/subsys/dce/sec**
   - **DFSDIR** as **/subsys/dce/dfs**

When you install a component, **dce_config** installs only those binaries required for the selected component.

7.2.1 Beginning the Installation

To begin the installation, perform the following steps:

1. Select `1. INSTALL` from the DCE Main Menu.

   The **dce_config** script prompts for whether you are installing from the file system or from a media device.

   ```
   Location of DCE Binaries ( on node_name )


           1. Filesystem
           2. Media


          selection:
   ```

   At this point, you have two choices:

   a. Type **1** and press <**Return**> to install the binaries from the install tree on the file system.

      The **dce_config** script then displays:

      ```
      Enter the full path to the DCE binary install tree.
       This will be the directory that contains the
       .../opt/dce1.1 directory:
      ```

      Type the full pathname to the install tree and press <**Return**>.

   b. Type **2** and press <**Return**> to install the binaries from the install tree on a media device.

      The **dce_config** script then displays:

      ```
      Enter name of media device (/dev/rmt0):
      ```

      Type the name of the media device that contains the install tree and press <**Return**>.

After you have specified the location of the install tree, **dce_config** displays:

```
By default, the DCE binaries will be COPIED from the
install tree into /opt/dcelocal/bin.  In order to save
save space, you can choose to simply symlink them instead.

 1. Copy
 2. Symlink
```

2. Type **1** to copy the binary files to your node or **2** to access the binary files through symbolic links and press **<Return>**.

   The **dce_config** script displays the DCE Installation Menu:

```
DCE Installation Menu ( on node_name )

1. Security Server
2. CDS Server
3. DTS Server
4. GDS Server
5. DFS Server

6. DCE Client
7. Application Development Environment
8. Optional Utilities
9. Replica Security Server
10. DFS Client

98. Return to previous menu
99. Exit

selection:
```

From the DCE Installation Menu, you can choose to install any of the following items by typing the number associated with your selection at the `selection` prompt:

- The master Security server binaries and the CDS, DTS, GDS, and DFS server binaries
- The replica Security server binaries, the DFS client binaries, and all other components' client binaries

- The IDL compiler and header files for use in DCE application development
- The following optional utility:
  - **cdsbrowser**—A a tool for viewing the content and structure of the namespace that runs under windowing software based on the OSF/Motif

Note: From the DCE Installation Menu, you can enter multiple selections separated by spaces. For example to install the Security server, the CDS server, and the DFS server, you would enter **1 2 5**. This function can be especially useful if you have set the environment variables described in Chapter 10.

The remainder of this subsection first describes the prompts that you may receive during the installation of all DCE components and how to answer those prompts. After these descriptions, the steps to perform each type of installation available from the DCE Installation Menu are described.

7.2.2 Installation Prompts

The **dce_config** script prompts for information common to all component installations. If you are installing DCE for the first time, you will receive all these prompts. If you have completely removed a previous installation or if you have continued an installation in a new **dce_config** session, you will receive only some of the prompts.

The common prompts you can receive are:

- A prompt to remove all previous DCE installations
- A prompt for message catalog locations
- If your machine has an existing **zoneinfo** directory, a prompt to remind you create a link to the new **zoneinfo** directory
- A prompt for the machine's ETHERNET address
- A prompt for whether or not to execute the OSF/1 **lib_admin** tool

Each prompt is described in the following subsections.

7.2.2.1 Removing All Previous Installations

During installation you will receive the following prompt:

```
Do you wish to first remove all remnants of
previous DCE installations for all components?
You should do so if you plan on re-installing
all existing DCE components now (n):
```

You will receive this prompt only once per **dce_config** session. This means that if you exit **dce_config** and then start it again, you will again receive this prompt.

To answer the prompt, you have two choices:

1. Type **n** and press **<Return>** if any of the following are true:
   - This is the first time you have installed DCE on the machine or you have removed the previous binaries with the **REMOVE** option
   - You are reinstalling only some of the components
2. Type **y** and press **<Return>** if you are reinstalling all components. Entering **y** is equivalent to choosing **REMOVE** from the DCE Main Menu.

After you answer the prompt, **dce_config** continues the installation.

7.2.2.2 Specifying the Message Catalog Location

During installation, you may receive the following prompt for the location of the DCE message catalogs:

```
Enter the directory into which message catalogs
should be stored on the local machine
(/usr/lib/nls/C):
```

You will receive this prompt under any of the following conditions:

- If you are installing DCE for the first time
- If you have used the **dce_config REMOVE** option to stop DCE daemons and remove any data files associated with those daemons.

To answer this prompt, type the full pathname name of the directory in which the DCE message catalogs should be stored and press **<Return>**. Alternatively you can simply press **<Return>** to accept the default. Note that the actual default directory depends on the operating system on your machine.

7.2.2.3 Verifying the zoneinfo Directory

During installation, you will receive the following prompt if your machine has an existing **/etc/zoneinfo** directory:

```
The DCE version of the zoneinfo data files has been
installed in /opt/dcelocal/etc/zoneinfo.  You may need
to replace /etc/zoneinfo with a link to this directory
for the DCE time component to work correctly.
Press <RETURN> to continue, CTRL-C to exit:
```

This prompt is to remind you that you must replace the existing **zoneinfo** directory with a link to the newly installed DCE **zoneinfo** directory. You can do this after the installation completes.

To continue the installation, press **<Return>** at the prompt.

7.2.2.4 Verifying the Ethernet Address

During installation, you may receive the following prompt to verify the machine's 12-digit hexadecimal Ethernet address:

```
The following IEEE 802 address has already
been entered for this machine:
0000c0eac744

Is this correct? (y)
```

You will receive this prompt under any of the following conditions:

- If you are installing DCE for the first time

- If you have used the **dce_config REMOVE** option to stop DCE daemons and remove any data files associated with those daemons.

To answer the prompt, you have two choices:

- If the displayed address is correct, press <**Return**>.
- If the displayed address is incorrect,
  1. Type **n** and press <**Return**>.

     The **dce_config** script prompts for the correct address.

     ```
     Enter your machine's IEEE 802 address as a
     12 digit hex number (for example, 08002B0F59BB):
     ```

  2. Type the correct address and press <**Return**>.

The **dce_config** script continues the installation.

Note: If **dce_config** cannot obtain the machine's Ethernet address, it will prompt for the address without displaying an address to verify.

7.2.2.5 Installing OSF/1 Shared Libraries — For OSF/1 Machines Only

If you are installing DCE on an OSF/1 platform, the **dce_config** script prompts for whether or not to install **lib_admin**, a tool to install and load the OSF/1 shared libraries, by displaying:

```
Run /sbin/lib_admin? (y)
```

Type **y** to install the OSF/1 shared libraries or **n** to not install the libraries and press <**Return**>. **dce_config** continues with the installation. If you plan to configure DCE, you should answer yes to this question.

7.2.3 Performing the Installations

The following subsections describe each type of installation that can be performed from the DCE Installation Menu. For all installations, except DFS servers, optional utilities, and DFS clients, the installation steps consist mainly of selecting the installation from the DCE Installation Menu and answering any of the common prompts described in the previous subsection.

The description of installing the Security server in the subsection that follows shows the steps involved if each of the common prompts were to be displayed. The descriptions of the other available installations do not show the common prompts.

7.2.3.1 Installing the Master Security Server

You must install one master Security server in each cell. Although you can have only one master Security server in each cell, you can install any number of replica Security servers. (See "Installing the Security Server Replicas" for more information.) The machine on which you install the master Security server must be reliable and physically secure.

In order to show all the common prompts you could receive, the installation steps in this subsection assume that this is the first time you have installed the DCE. When you install a Security server, you may not receive all the prompts shown.

To install the master Security server, perform the steps listed below:

1. At the DCE Installation Menu, type **1** and press <**Return**>.

   The **dce_config** script asks whether or not you want to remove the previous DCE installation by displaying the following prompt:

   ```
   Do you wish to first remove all remnants of
   previous DCE installations for all components?
   You should do so if you plan on re-installing
   all existing DCE components now (n):
   ```

2. Because you are installing the master Security server for the first time, type **n** and press <**Return**>.

The **dce_config** script installs the master Security server and the DCE client binaries, displaying what it is doing as it does it.

Then, if you are installing on an OSF/1 machine, **dce_config** prompts for whether to install **lib_admin**, a tool to load and install and load the OSF/1 shared libraries:

```
Run /sbin/lib_admin? (y)
```

3. Type **y** to install the OSF/1 shared libraries or **n** to not and press <**Return**>.

   The **dce_config** script then prompts for the location of the message catalogs:

   ```
   Enter the directory into which message catalogs
   should be stored on the local machine
   (/usr/lib/nls/C):
   ```

4. Type the full pathname name of the directory in which the DCE message catalogs should be stored and press <**Return**>. Alternatively you can simply press <**Return**> to accept the default. Note that the actual default directory depends on the operating system on your machine.

   If your machine has an existing **/etc/zoneinfo** directory, **dce_config** displays the following prompt:

   ```
   The DCE version of the zoneinfo data files has been
   installed in /opt/dcelocal/etc/zoneinfo.  You may need
   to replace /etc/zoneinfo with a link to this directory
   for the DCE time component to work correctly.
   Press <RETURN> to continue, CTRL-C to exit:
   ```

   This prompt is to remind you that you must replace the existing **zoneinfo** directory with a link to the newly installed DCE **zoneinfo** directory. You can do this after the installation completes.

5. To continue the installation, press <**Return**> at the prompt.

   If you are installing DCE on the OSF/1 reference platform and the **/etc/ieee_802_addr** file exists, the **dce_config** script prompts you to verify the machine's 12-digit hexadecimal Ethernet address:

```
The following IEEE 802 address has already
been entered for this machine:
0000c0eac744

Is this correct? (y)
```

The **dce_config** script takes the displayed address from the Ethernet card or from the **/etc/ieee_802_addr** file. If the script cannot obtain the Ethernet address, it will prompt for it.

At this point, you have two choices:

a. Press **<Return>** if the displayed address is correct.

b. If the displayed address is incorrect, type **n** and press **<Return>**. The **dce_config** script prompts for the correct address.

   ```
   Enter your machine's IEEE 802 address as a
   12 digit hex number (e.g. 08002B0F59BB):
   ```

   Type the correct address and press **<Return>**.

The **dce_config** script completes the installation and returns the DCE Installation Menu.

7.2.4 Installing the CDS Servers

You must install and configure at least one CDS server in each cell. For performance reasons, you may install and configure more than one. To install a CDS server, type **2** at the DCE Installation Menu and press **<Return>**.

7.2.5 Installing DTS Servers

It is recommended that you run at least three DTS servers in each cell and that you configure at least one of those servers with a DTS Time Provider as described in Chapter 8.

To install a DTS server, type **3** at the DCE Installation Menu and press **<Return>**.

7.2.6 Installing a GDS Server

To install a GDS server, type **4** at the DCE Installation Menu and press **<Return>**. Note that when you install a GDS server, **dce_config** automatically installs the DCE client binaries if necessary.

7.2.7 Installing the DFS Servers

To install the DFS Servers, perform the following steps:

1. At the DCE Installation Menu, type **5** and press **<Return>**.

   The **dce_config** script displays the DFS Server Installation menu:

   ```
   Four types of DFS Server installations are valid.

   1.  System Control Machine
   2.  Private File Server
   3.  File Server
   4.  Fileset Location Database Server

   selection:
   ```

2. Select the type of server you want to install by typing its associated selection number, and pressing **<Return>**.

   The steps to perform the installation and the prompts you must answer for each type of DFS server are the same. The following steps show the installation of a File Server Machine.

   After you select a server installation from the DFS Server Installation menu, the **dce_config** script proceeds to install the server. Then, **dce_config** prompts for whether to install the optional DFS servers:

```
Optional DFS servers are:
    cm fms udebug
    scout upclient upserver
Would you like to install the optional DFS servers? (y):
```

3. Type **y** to install the optional servers or **n** to not install them and press <**Return**>.

   Note that when you install a DFS server, **dce_config** automatically installs the DCE client binaries.

   The **dce_config** script then prompts for whether to install additional optional command client binaries on the machine. Generally it is useful to install these administrative commands, particularly the **cm** command that is used to manage the client cache manager.

   ```
   Would you like to install: cm bos fts bak,
   (all are optional) on this machine (y)?
   ```

4. Type **y** to install the optional binaries or **n** to not install them and press <**Return**>.

   When **dce_config** completes the DFS installation, it returns the DCE Installation Menu.

7.2.8 Installing a DCE Client

Since all machines that will engage in network communications must be DCE clients, you should install the DCE client binaries on every machine in the cell. Note that the DCE client binaries are automatically installed on every machine on which you have installed a DCE server during the installation. Therefore, you must install the client binaries only on those machines on which you have not installed a server.

To install the client binaries, type **6** at the DCE Installation Menu and press <**Return**>.

The **dce_config** script completes the DCE client installation and returns the DCE Installation Menu.

7.2.9 Installing The Application Development Environment

The Application Development Environment installation sets up the **.idl** and **.h** files used for DCE application development.

To install the Application Development Environment **.idl** and **.h** files, type **7** at the DCE Installation Menu.

The **dce_config** script displays the files it installs them. Note that all the files except the pthread files are installed in **usr/include/dce**. Pthread files are installed in **usr/include**. Note that if your machine has existing files (such as **pthread.h**) in **usr/include**, you should replace those files with links to the DCE versions. As it installs the files, **dce_config** will display warning messages detailing which files should be replaced with links.

When the installation is complete, **dce_config** returns the DCE Installation Menu.

7.2.10 Installing the Optional Utilities

You can install the following optional utilities:

- **cdsbrowser**—A a tool for viewing the content and structure of the namespace that runs under windowing software based on the OSF/Motif

To install the optional utilities, perform the following steps:

1. Type **8** at the DCE Installation Menu and press **<Return>**.

   The **dce_config** script displays the Optional Utilities Installation menu.

```
Optional Utilities Installation Menu

1. cdsbrowser

98. Return to previous menu
99. Exit

selection:
```

2. Type the number associated with the optional utility you want to install and press **<Return>**.

   The **dce_config** script installs the chosen utility, displaying a message describing what it is doing as it does it. When installation is complete, **dce_config** returns the Optional Utilities Installation menu.

7.2.11 Installing a Security Server Replica

You can install replicas of Security servers within your cell to help ensure the availability of the registry database. To install a replica Security server, type **9** at the DCE Installation Menu and press **<Return>**.

The **dce_config** script installs the replica, displaying a message describing what it is doing as it does it, and returns the DCE Installation Menu.

7.2.12 Installing DFS Clients

DFS client binaries are automatically installed on those machines on which you have installed a DFS server. Therefore, you must install the client binaries only on those machines on which you have not installed a server.

To install the DFS client binaries, perform the following steps:

1. Type **10** at the DCE Installation Menu and press **<Return>**.

   The **dce_config** script installs the client binaries and prompts for whether to install additional optional command client binaries on the machine. Generally it is useful to install these administrative commands, particularly the **cm** command that is used to manage the client cache manager.

   ```
   Would you like to install: cm bos fts bak,
   (all are optional) on this machine (y)?
   ```

2. Type **y** to install the optional servers or **n** to not install them and press **<Return>**.

When **dce_config** completes the DFS client installation, it returns the DCE Main Menu.

Chapter 8

Configuring DCE

This chapter describes how to use the **dce_config** script to configure DCE software once that software has been installed.

Note: Because of differences in DCE R1.1 platforms, the output in the sample procedures in this chapter may differ slightly from the output you see when you run the **dce_config** script.

This chapter also describes how to use the Code Set Registry Compiler (**csrc**) to build a code set registry once DCE software has been configured. This step is only necessary if you are configuring an internationalized DCE cell, which is a cell that supports RPC applications that use a wide variety of languages other than U.S. English.

8.1 Prerequisites

Before you can configure DCE software, you must have installed it on each machine that you are configuring. See Chapter 7 for instructions on installing the DCE software.

You must run **dce_config** on the machine you are configuring. You cannot configure a machine remotely. Additionally you must be logged into the local machine as root.

8.2 Order of Configuration

You must configure machines in a cell in the following order:

1. The master Security server
2. The initial CDS server
3. A DTS server
4. A single DTS time provider
5. DCE client machines
6. DFS servers, Security and CDS replicas, GDA servers, Password Management servers, and additional time servers

Note also that you must configure a CDS client on all Security server (master or replica) machines that are not running a CDS server. You must also configure a Time client on all machines that are not running a Time server. Be sure to configure the clients only after you have configured all servers.

8.3 Split Server Configurations

If you configure the master Security server and the CDS initial server on different machines, the cell is said to have a split server configuration. If you choose a split server configuration, you must take some extra steps in the initial cell configuration phase. This is because every DCE machine must be configured as a DCE client, including machines that are also configured as some kind of DCE server machine. However, a DCE client configuration cannot succeed unless a CDS server is available. Therefore, the initial Security server machine cannot immediately be configured as a DCE client, but must wait until a CDS server machine is configured.

This is not a problem when the master Security server and CDS server are configured on one machine. However, in a split server configuration, you must first configure the Security server, then configure the CDS server on another machine, and then

come back to the Security server machine and configure it as a DCE client or as an additional server. The following steps detail these actions:

1. Configure Machine 1 as a Security server machine.
2. Go to Machine 2 and configure it as a CDS server machine.
3. Go back to Machine 1 and configure it as a DCE client.

Note that if you configure the Security server machine as some other kind of server machine (for example, as a DTS server), then it is configured as a DCE client. So the following scenario also works:

1. Configure Machine 1 as a Security server machine. The **dce_config** script prompts you with the clock synchronization question discussed in the following subsection. When it does this, answer **n**.
2. Go to Machine 2 and configure it as a CDS server machine.
3. Go back to Machine 1 and configure it as a DTS server.

In this scenario, Step 3 causes Machine 1 to be configured as a DCE client as well as a DTS server.

Consult the *OSF DCE Release Notes* for the current information on timing and sequence constraints when bringing up split server cells.

8.4 Clock Synchronization

All servers and clients being configured should have clocks that are at least loosely synchronized. Using the **SYNCH_CLOCKS** environment variable described in Chapter 10, you can specify that the clocks be synchronized automatically or that the user should be prompted to synchronize the clocks if they are out of synch by a specified amount. For DFS configurations, the clocks should be synchronized automatically.

While it is not required, it is recommended that you configure the Security master server machine with a DTS server. This allows Security clients to synchronize clocks with the Security server machine.

8.5 Security and CDS Database Size

Both the Security servers (master and replicas) and the CDS servers (initial and replicas) maintain databases. The machine on which a Security server or a CDS server is configured must have sufficient disk space to accommodate these databases. The size required for the Security and CDS databases depends on the platform and the operating system. Choose machines large enough to accommodate future growth.

Use the following general requirements for the Security database:

- For each principal: 1000 bytes of physical memory and 700 bytes of disk space.
- For each account: 1700 bytes of physical memory and 300 bytes of disk space.

The virtual size of the CDS databases increases substantially as new directories and objects are added. When entries are removed, the memory associated with them may be reused but is not deallocated. Although it can vary depending on the platform, each directory can use roughly 38.5 Kbytes of virtual space. Process data size and paging space should also be considered.

8.6 Accessing the DCE Configuration Menu

All configuration functions are chosen from the DCE Configuration Menu. To access the DCE Configuration Menu, at the DCE Main Menu, type **2** and press **<Return>**.

The **dce_config** script displays the DCE Configuration Menu.

```
DCE Configuration Menu (on host_name)

  1. Initial Cell Configuration
  2. Additional Server Configuration
  3. DCE Client
  4. DFS Client

 98. Return to previous menu
 99. Exit
```

```
selection:
```

From the DCE Configuration Menu, you can choose to configure any of the following items by typing the number associated with your selection at the `selection` prompt:

- Selection **1**, `Initial Cell Configuration`, lets you configure the master Security server, the initial CDS server, and DTS servers.
- Selection **2**, `Additional Server Configuration`, lets you configure the following servers:
  - — Replicas of the master Security server
  - — Replicas of the initial CDS server
  - — DTS servers
  - — DFS System Control Machine
  - — DFS Private File server
  - — DFS File server
  - — DFS Fileset Location Database server
  - — GDA server
  - — Password Management Server
- Selection **3**, `DCE Client`, lets you configure DCE client machines for all components except DFS.
- Selection **4**, `DFS Client`, lets you configure DFS client machines.

The remainder of this subsection takes you through the steps to perform each type of configuration available from the DCE Configuration Menu. Note that in the sample output, the **DCE_DISPLAY_THRESHOLD** environment variable is set to the default, **SUMMARY**, therefore **dce_config** displays only Error, Warning, and Summary messages.

See Chapter 10 for more information about controlling message logging.

8.6.1 The Initial Privileged User

When you configure the master Security server, you create an account for the initial privileged user of the registry database, login to that account, and are authenticated as the privileged user. If you exit from the **dce_config** script, re-invoke it, and choose `Configure` from the main menu, you will be prompted to log in as this privileged user. Additionally, at other points in the configuration process, you may be prompted to reauthenticate as the privileged user.

8.6.2 Specifying the Removal of Previous Configurations

When you configure your cell initially, by selecting items from the Initial Cell Configuration menu, or when you configure DCE clients or DFS clients, you may receive the following prompt:

```
Do you wish to first remove all remnants of
previous DCE configurations for all components (y/n)?
You should do so if you plan on re-configuring
all existing DCE components now.
```

This prompt appears once in each **dce_config** session. This means that once you answer it, it will not appear again unless you exit **dce_config** and then restart **dce_config**. The prompt asks whether or not you want to remove the previous DCE configurations.

To answer the prompt, you have two choices:

- Type **n** and press **<Return>** if any of the following are true:
  - — This the first time you have configured DCE on the machine or you removed the previous configuration with the **REMOVE** option
  - — If you are reconfiguring only some of the components
- Type **y** and press **<Return>** if you are reconfiguring all components.

  If you enter **y**, **dce_config** stops all DCE daemons and deletes any data files associated with the daemons. It displays messages telling you what it is doing while it does it.

8.7 Performing Initial Cell Configuration

This section describes the tasks to configure the master Security server, the initial CDS server, and DTS time servers and time providers. These servers are the basic servers required for any cell. You must configure the master Security server and the initial CDS server before you can configure client machines, any DFS servers, and Security and CDS replicas. See ''Configuring Additional Servers'' for information on configuring client machines, DFS servers, and Security and CDS replicas.

In the subsections that follow, all procedures start at the Initial Cell Configuration menu.

8.7.1 Accessing the Initial Cell Configuration menu

All initial cell configuration functions are accessed from the Initial Cell Configuration menu. To access the Initial Cell Configuration menu, at the DCE Configuration Menu, type **1** and press **<Return>**.

The **dce_config** script displays the following message:

```
S:****** Configuring initial cell...
```

and then the Initial Cell Configuration menu.

```
Initial Cell Configuration ( on host_name )

  1. Initial Security Server
  2. Initial CDS Server
  3. Initial DTS Server

 98. Return to previous menu
 99. Exit

 selection:
```

Note: You can enter more than one selection at a time from the DCE Configuration Menu. Just be sure to separate the selections by spaces. For example to configure the master Security server and the initial CDS server, you would

enter **1 2**. This function can be especially useful if you have set the environment variables described in Chapter 10.

8.7.2 Configuring the Master Security Server

You must configure the master Security server before any other machine in the cell. As part of the master Security server configuration, **dce_config** creates the cell's initial privileged user and the registry database, which will contain users and accounts. Once the master is in place, you can create Security server replicas on other machines.

To configure the master Security server:

1. At the DCE Initial Cell Configuration menu, type **1** and press <**Return**>.

   The **dce_config** script displays the following message:

   ```
   S:****** Configuring initial Security Server...
   ```

   The **dce_config** script prompts for the name of the cell in which the servers are being configured:

   ```
   Enter the name of your cell (without /.../):
   ```

2. Type the name of the cell and press <**Return**>.

   The **dce_config** script displays the following message:

   ```
   S:****** Starting dced...
   ```

   The **dce_config** script displays the following prompt:

   ```
   Enter keyseed for initial database master key:
   ```

3. Type in the text string for the keyseed, which is a temporary DES key that is used to generate the registry's master key (the key that the registry will use for account key creation). Press <**Return**>.

   The text you enter should not be easily guessed. Note that it is not displayed as you type it.

The **dce_config** script prompts:

```
Enter desired principal name for the Cell Administrator: (cell_admin)
```

Type the name of the principal who will be the initial privileged user of the registry database (known as the registry creator) and press <**Return**>. You will use the account created by **dce_config** for the principal you name here to log in to the DCE during subsequent component configurations.

The **dce_config** script prompts:

```
Enter desired password for the Cell Administrator:
```

4. Type the password to be assigned to the initial privileged user account and press <**Return**>. Note that if you use the default password, **-dce-** you should change it to a more secure password after DCE configuration is complete.

   The **dce_config** script prompts for the password again to ensure it is correct:

   ```
   Re-enter desired password:
   ```

5. Type the password again and press <**Return**>.

   The **dce_config** prompts:

```
S:****** The current highest UNIX ID for persons on this node is 30124.
        Enter the starting point to be used for UNIX ID's that
        are automatically generated by the Security Service
        when a principal is added using "rgy_edit": (30224)
```

6. Type a UNIX ID number that will be used as the principal UNIX ID at which the Security server will start assigning automatically generated principal UNIX IDs and press <**Return**>. The default is the value of the highest principal UNIX ID on the machine being configured, incremented by the value of the **UID_GAP** environment variable. Although the value you supply is not required to be higher than the machine's highest principal UNIX ID, if you supply a value that is less than or equal to the highest currently used principal UNIX ID, **dce_config** issues a warning message and prompts you to reenter the UNIX ID.

   The **dce_config** script prompts:

```
S:****** The current highest UNIX ID for groups is 26.
        Enter the starting point to be used for UNIX ID's that
        are automatically generated by the Security Service
        when a group is added using "rgy_edit": (126)
```

7. Type a UNIX ID number that will be used as the group UNIX ID at which the Security server will start assigning automatically generated group UNIX IDs and press <**Return**>. The default is the value of the highest group UNIX ID on the machine incremented by the value of the **UID_GAP** environment variable. Although the value you supply is not required to be higher than the machine's highest group UNIX ID, if you supply a value that is less than or equal to the highest currently used group UNIX ID, **dce_config** issues a warning message and prompts you to reenter the UNIX ID.

   The **dce_config** script starts the **secd** server and initializes the registry database. As it does, it displays:

   ```
   S:****** Starting secd...
   S:****** Checking for active sec_client service...
   S:****** Initializing the registry database...
   ```

   When the master Security server configuration is complete, **dce_config** returns the Initial Cell Configuration menu.

8.7.3 Configuring the Initial CDS Server

The initial CDS server must in place for cell operation and before any of the cell's client machines are configured. Other additional CDS servers can configured on client machines.

To configure the Initial CDS server:

1. At the Initial Cell Configuration menu, type **2** and press <**Return**>.

   The **dce_config** script displays the following messages:

   ```
   S:****** Configuring initial CDS Server...
   S:****** Checking for active sec_client service...
   ```

Then, the **dce_config** script prompts:

```
Create LAN profile so clients and servers can be
divided into profile groups for higher performance in a multi-LAN cell? (n)
```

2. At this point you have two choices:

   a. Type **y** to configure the machine with multiple local area network (LAN) capabilities and press **<Return>**. You may want to use multiple LANs to organize how applications find objects in the namespace. When you respond **y**, the **dce_config** script creates a LAN profile that allows the DTS clerks to synchronize with DTS servers on the local LAN.

      The **dce_config** script prompts for the name of the LAN:

      ```
      What is the name of the LAN?
      ```

      Type the name and press **<Return>**. You can enter any arbitrary name. The name is used by **dce_config** to store cell profile information.

   b. Type **n** to not configure the machine with multiple LAN capabilities and press **<Return>**.

   The **dce_config** script completes the installation, starts the **cdsadv** and **cdsd** servers, creates the LAN profile (if necessary), and sets the appropriate ACLs. It displays the actions it takes as it takes them.

   When the configuration is complete, **dce_config** returns the DCE Initial Cell Configuration Menu.

8.7.4 Configuring a DTS Server

A DTS Clerk should be configured on each machine in a cell that consists of more than one machine. A minimum of three DTS servers is recommended for any cell containing three or more machines.

Note that you can use DTS commands to reconfigure DTS after the initial configuration is complete. See the *OSF DCE Administration Guide—Core Components* for more information.

You can configure two types of DTS machines:

- **Clerks**—DTS clerks receive time values and adjust the system clock accordingly. Most DTS daemons are configured as clerks.
- **Servers**—DTS servers synchronize times in the cell in addition to performing DTS clerk tasks. You can configure two types of servers: global servers for cells with multiple LANs and local servers for cells without multiple LANS.

You can optionally configure DTS servers with a time provider. A time provider obtains the time either from the system clock on the machine on which the server is running a null time provider or from an outside time source, such as radio, telephone, or satellite (an NTP time provider). The NTP time provider interfaces with a host running the NTP time service, which in turn should access a reliable time source.

You configure your cell's DTS servers first, then designate one of them as the location of a time provider, as well. Section 8.7.4.3 provides instructions for configuring a null or NTP time provider on a server machine.

8.7.4.1 Accessing the DTS Configuration Menu

DTS configuration is performed from the DTS Configuration Menu. To access this menu, at the Initial Cell Configuration menu, type **3** and press **<Return>**.

The **dce_config** script displays the following messages:

```
S:****** Configuring initial DTS services
S:****** Please wait for user authentication and authorization...
```

and then the DTS Configuration Menu.

```
DTS Configuration Menu

  1. DTS Local Server
  2. DTS Global Server (needed only in multi-LAN cells)
  3. DTS Clerk
  4. DTS Time Provider

  98. Return to previous menu
```

```
99. Exit

selection:
```

8.7.4.2 Configuring Local Servers, Global Servers, and Clerks

You can configure a single machine either as a server (local or global) or as a clerk, but not both. To configure a local server, global server, or clerk, type the appropriate selection number at the DTS Configuration Menu and press **<Return>**.

If you configured a local server, **dce_config** displays:

```
S:****** Configuring DTS Local Server...
S:****** Waiting for node self-identity to be established...
S:****** Starting dtsd...
S:****** This node is now a DTS local server.
```

If you configured a global server, **dce_config** displays:

```
S:****** Configuring DTS Global Server...
S:****** Waiting for node self-identity to be established...
S:****** Starting dtsd...
S:****** This node is now a DTS global server.
```

If you configured a Clerk, **dce_config** displays:

```
S:****** Configuring DTS Clerk...
S:****** Starting dtsd...
S:****** This node is now a DTS clerk.
```

When the configuration is complete, **dce_config** returns the DTS Configuration Menu.

8.7.4.3 Configuring a Time Provider on a DTS Server

You should configure one of the DTS servers in the cell with a provider of the accurate time. You can configure two types of time providers: a null time provider and a NTP time provider.

To configure a DTS server as a time provider, perform the following steps:

1. Type **4** on the DTS Configuration Menu and press **<Return>**.

   The **dce_config** script displays The DTS Time Provider Menu.

   ```
   DTS Time Provider Menu

   1. Configure a NULL time provider
   2. Configure an NTP time provider

   98. Return to previous menu
   99. Exit

   selection:
   ```

2. Your next action depends on the kind of time provider you are configuring.

   a. To configure the server as a null time provider, type **1** and press **<Return>**.

      The **dce_config** script configures the server and returns DTS Configuration Menu.

   b. To configure the server as an NTP time provider, type **2** and press **<Return>**.

      dce_config prompts:

      ```
      Enter the hostname where the NTP server is running:
      ```

      Enter the full pathname to the host and press **<Return>**.

      The **dce_config** script configures the time provider and returns the DTS Configuration Menu.

8.8 Modifying ACLs on the Master Security Server

Upon initial cell creation, security is strong and little remote access is allowed — ACL settings on **dced** objects are tight. To allow for more flexibility in remote administration, you may want to modify the ACLs on certain **dced** objects. As root, run the **dced_acl_patcher** script on the master Security Server. This opens the ACL settings, while still preventing the **cell_admin** from having root access to all machines in the cell via **dced**.

To modify the ACLs, follow these steps:

1. Log in as root on the master Security Server.
2. Kill **dced** and patiently wait for it to terminate — termination can take up to several minutes.

   % kill -s TERM *pid*

3. Restart **dced** with the **-r** option:

   % dced -r

   Wait 30 seconds for **dced** to start.
4. Run the **dced_acl_patcher** script, which does not prompt you for information:

   % dced_acl_patcher

See Appendix A, for a comparison of the ACLs as they exist before and after running **dced_acl_patcher**.

8.9 Configuring Additional Servers

Using the **dce_config** script, you can configure the following additional servers:

- Additional DTS servers
- DFS servers

- GDA servers
- Security server replicas
- CDS server replicas
- Password Management server

Before you configure additional servers ensure that:

- The cell's master Security server and initial CDS server have been configured and started.
- The machine on which you are configuring additional servers has been configured as a DCE client.
- You are able to login as the initial privileged user of the Security database.

8.9.1 Accessing the Additional Server Configuration menu

All procedures to configure additional servers start at the Additional Server Configuration menu. To access this menu, at the DCE Configuration Menu, type **2** and press **<Return>**.

The **dce_config** script displays the following messages:

```
S:****** Configuring additional server...
S:****** Please wait for user authentication and authorization...
```

and then the Additional Server Configuration menu.

```
        Additional Server Configuration ( on host_name )


         1.  Additional CDS Server(s)
         2.  DTS
         3.  DFS System Control Machine
         4.  DFS Private File Server
         5.  DFS File Server
         6.  DFS Fileset Location Database Server
         7.  GDA Server
```

```
 8.  Replica Security Server
 9.  Auditing
 10. Password Management Server
 11. Unconfigure Password Management Server



98. Return to previous menu
99. Exit



selection:
```

Note that if you have not been authenticated as the initial privileged user of the registry database in the current **dce_config** session, **dce_config** prompts you for the privileged user's name and password before it displays the Additional Server Configuration menu.

The following subsections describe the steps to configure each type of additional server in order as they appear on the Additional Server Configuration menu.

8.9.2 Configuring Additional CDS Servers

When you configure additional CDS servers, you create replicas of the clearinghouse database created during the configuration of the initial CDS server. Replicas are useful to ensure the availability of name information when the machine on which the initial CDS server runs is unavailable. A CDS server replica must be created on a different machine than the one on which the master CDS server resides.

To configure the additional CDS servers:

1. At the Additional Server Configuration menu, type **1** and press **<Return>**.

   Note: If you attempt to configure an additional CDS server on the node where the master CDS server is located, the **dce_config** script returns and error message, and you must start again.

   The **dce_config** script displays the following messages:

```
S:******  Configuring additional CDS server.
S:******  Waiting for node self-identity to be established...
```

Then the **dce_config** script starts the CDS client daemon, and prompts:

```
What is the name for this clearinghouse?
```

2. Type the name to be assigned to the CDS clearinghouse and press <**Return**>. (The name assigned to the initial clearinghouse is in the form *machine_name***_ch**. You may want to continue this convention in naming clearinghouse replicas.)

   Note: Existing documentation and **dcecp** task scripts assume that you will follow this naming convention.

   The **dce_config** script displays the following messages:

```
S:******  Initializing the name space for additional CDS
          Server...
S:******  Setting ACLs for the CDS clearinghouse /.:/host_ch...
```

   and then prompts:

```
Should more directories be replicated? (n)
```

3. At this point, you have two choices:

   a. Type **y** to replicate other directories on the machine and press <**Return**>.

      The **dce_config** script prompts for a list of the directories to replicated:

```
Enter a list of directories to be replicated, separated
by spaces, and terminated by <Return>:
```

      Type in the list of directories separated by spaces. Press <**Return**> when the list is complete.

      The **dce_config** script automatically replicates the root directory. To have **dce_config** replicate other directories, you must specify the full pathname (including the **/.:/**) of each additional directory you want to replicate. Specifying the root directory of a tree structure does not cause the entire tree

structure to be replicated. (Note that once the configuration is complete, you can use the CDS control program (**cdscp**) to replicate directories at any time.)

b. Type **n** to not replicate directories and press <**Return**>.

The **dce_config** script completes the configuration and returns the Additional Server Configuration menu.

4. When the additional server configuration is complete, exit **dce_config** and log in as the initial privileged user (created during the configuration of the master Security server), and skulk the root directory using the following command:

```
set directory /.: to skulk
```

This action propagates a copy of the changed root directory information to all the CDS servers. Note that use of several CDS servers may expand the time required to complete the propagation.

8.9.2.1 Possible Configuration Errors

An additional CDS server configuration may fail if the **cdsd** has been successfully launched, but its clearinghouse has not been properly created. Such an event may produce error messages like the following:

```
ERROR:   Error during creation of clearinghouse /.:/nodename_ch.
           Message from cdscp:
           Failure in routine: cp_create_clh; code = 282109010
           Requested operation would result in lost connectivity
           to root directory (dce / cds)

ERROR:   Error during creation of clearinghouse /.:/nodename_ch.
           Message from cdscp:
           Failure in routine: cp_create_clh; code = 282108908
         Unable to communicate with any CDS server (dce / cds)
```

8.9.2.2 Handling Configuration Errors

If you receive either of these messages, the clearinghouse is in an intermediate state and cannot be used or deleted, although the rest of the cell namespace and other servers are unaffected. To recover:

1. Skulk the root directory
2. Use the **create clearinghouse /.:/***clearinghouse_name* command while you are logged in as the initial privileged user on the newly configured CDS server machine. This command manually completes the configuration of the new server and its clearinghouse.
3. Finally, skulk the root directory again.

In rare circumstances, you may see the following error, which you can ignore.

```
ERROR: cdscp error during "define cached server" command.
       Message from cdscp:
       Failure in routine: cp_define_cached_server;
       code = 282111142
       Cached Server clearinghouse already exists (dce / cds)
```

The error results from **dce_config** trying to repeat an operation it has already performed.

8.9.3 Configuring Additional DTS Servers

The steps to configure additional DTS servers (selection 2 from the Additional Server Configuration menu) are the same as the steps to configure the initial DTS servers. See Section 8.7.4 for instructions.

8.9.4 Configuring DFS Servers

You can configure the following types of DFS servers:

- A System Control machine, which distributes system configuration information that is shared by all DFS server machines in a cell.

- A Fileset Location Database machine, which tracks and records the locations of all filesets. At least one is required in each cell. You can run multiple Fileset Location Database servers for to help ensure the server's availability at all times.
- A File Server machine, which exports LFS and non-LFS data to the global DFS namespace.
- A Private File Server, which also exports file system data to the global DFS namespace. A private File Server machine is controlled by the owner of the machine, not by the system administrator. The purpose of a private File Server machine is to allow individual users to export a small number of filesets. A Private File Server does not support replicated filesets.

Note: System Control machines can also be used as Binary Distribution machine.

Although you must configure at least one Fileset Location Database machine, all other DFS server machines are optional. Note that if your are configuring a System Control machine, you should configure it first because you are prompted to enter its name during the Fileset Location Database machine configuration. Otherwise, you should configure the Fileset Location Database machine first.

8.9.4.1 Prerequisites

Before you begin DFS configuration, be sure that you have:

- Made the planning decisions described in the *OSF DCE DFS Administration Guide and Reference*. These decisions include the following:
  - — Choosing the first DFS machine
  - — Determining the cache size on DFS client machines
  - — Defining DFS domains
  - — Setting up your cell's DFS tree structure
  - — Determining which file system to use for **root.dfs**
- Created the file system for the **root.dfs** fileset. This file system can be either a DCE LFS fileset or a non-LFS file system (non-LFS fileset). To use a non-LFS file system, create the file system manually by using your operating system's file system commands. To use a DCE LFS fileset, create a DCE LFS aggregate with the DFS **newaggr** command.

Before setting up the **root.dfs** fileset, refer to the *OSF DCE DFS Administration Guide and Reference*.

Note: The **dce_config** script configures the Fileset Location Database Server on the machine that exports **root.dfs**.

- On HP-UX machines, ensured that the kernel is installed on the machine being configured. This kernel, which is available on the installation tree, is named **hpux.dfs**. To install it, first rename the existing kernel(named **/hpux**) and then copy **hpux.dfs** to the / directory, naming it **hpux**.

  For other platforms you should run the DFS kernel that is built during the build process. (For OSF/1 machines, the **fsvmunix** kernel includes support for both DFS and DCE LFS. The **dfsvmunix** kernel includes support only for DFS, but requires less disk and memory resources.)

8.9.4.2 Configuring a System Control Machine

To configure a System Control machine, perform the following steps:

1. Type **3** on the DCE Configuration Menu and press <**Return**>.

   The **dce_config** script displays the following message:

   ```
   S:****** Configuring DFS System Control Machine...
   ```

   and then prompts:

   ```
   Enter Cell Administrator's principal name: (cell_admin)
   ```

2. Type the name of the principal who was defined to be the initial privileged user of the registry database during the configuration of the master Security server and press <**Return**>.

   The **dce_config** script prompts:

   ```
   Enter password:
   ```

3. Type the password for the initial privileged user's account and press **<Return>**. Note that this password is not displayed as you type it.

   The **dce_config** script loads the DFS kernel extensions and displays the following messages:

   ```
   S:****** Loading kernel extensions...
   rpc_config: installed krpc device at major number 71
   ```

   Then the **dce_config** script prompts for whether LFS (Episode) should be loaded

   ```
   Should the LFS Kernel Extension be loaded (n)?
   ```

4. Type **y** to configure the machine to use LFS or **n** to not and press **<Return>**.

   The **dce_config** script displays:

   ```
   Should LFS be initialized (n)?
   ```

5. Type **y** to initialize LFS or **n** to not and press **<Return>**.

   The **dce_config** script displays:

```
S:****** Modifying the registry database for DFS server operation...
S:****** Starting bosserver...
S:****** Creating BOS admin lists.
S:****** Starting upserver...
```

When the configuration is complete, **dce_config** displays the Additional Server Configuration menu.

8.9.4.3 Configuring a File Server and a Private File Server

The steps to configure a File Server and a Private File Server Server are the same. In the steps that follow, a Private File Server is configured to illustrate the sequence of prompts and actions. However, you can use the same instructions to configure a File Server.

To configure a File Server or Private File Server, perform the following steps:

1. Type **4** (for a Private File Server) or **5** (for a File Server) on the Additional Server Configuration menu and press <**Return**>.

   The **dce_config** script displays the following message:

   ```
   S:****** Configuring DFS Private File Server...
   ```

   and then prompts:

   ```
   Enter Cell Administrator's principal name: (cell_admin)
   ```

2. Type the name of the principal who was defined to be the initial privileged user of the registry database during the configuration of the master Security server and press <**Return**>.

   The **dce_config** script prompts:

   ```
   Enter password:
   ```

3. Type the password for the initial privileged user's account and press <**Return**>. Note that this password is not displayed as you type it.

   The **dce_config** script loads the DFS kernel extensions and displays the following message:

   ```
   S:****** Loading kernel extensions...
   ```

   Then the **dce_config** script prompts for whether LFS (Episode) should be loaded:

   ```
   Should the LFS Kernel Extension be loaded (n)?
   ```

4. Type **y** to configure the machine to use LFS or **n** to not and press <**Return**>.

   The **dce_config** script displays:

   ```
   Should LFS be initialized (n)?
   ```

5. Type **y** to initialize LFS or **n** to not and press <**Return**>.

The **dce_config** script displays:

```
S:****** Modifying the registry database for DFS server operation...
S:****** Starting bosserver...
S:****** Starting ftserver...
```

Then **dce_config** script prompts:

```
Enter the name of the system control machine:
```

6. Type the name of the machine configured as the system control machine and press **<Return>**. If you cell does not use a system control machine enter the name of the local machine.

   The **dce_config** script prompts:

```
        Enter the filesystem type for the aggregate to be exported:
        1. Native File System (e.g. UFS, JFS)
        2. Episode File System (LFS)
selection:
```

   Your next steps depend on whether you want to export the native file system or the Episode file system. If you are exporting the native file system, go to the subsection titled ''Exporting the Native File System.'' If you are exporting the Episode file system, go to the subsection titled ''Exporting the Episode File System.''

   Exporting the Native File System

   To export the native file system, perform the following steps.

   a. Type **1** and press **<Return>**.

      The **dce_config** script prompts:

```
Enter the device name for the aggregate to be exported (i.e. /dev/lvXX):
```

      Note: The information in parentheses in the previous prompt is not the default value, but only an example of a value to be entered.

b. Type the pathname of the device on which the selected file system is stored and press <**Return**>.

The **dce_config** script prompts:

```
Enter the mount path for the aggregate (e.g. /usr/users):
```

Note: The information in parentheses in the previous prompt is not the default value, but only an example of a value to be entered.

c. Type the pathname to the aggregate's mount point and press <**Return**>.

The **dce_config** script prompts:

```
Enter a unique aggregate name (e.g. user.jlw):
```

Note: The information in parentheses in the previous prompt is not the default value, but only an example of a value to be entered.

d. Type a name for the aggregate and press <**Return**>. The aggregate name must be unique on the machine being configured.

The **dce_config** script prompts:

```
Enter a unique numerical aggregate ID:
```

e. Type a number for the aggregate ID and press <**Return**>. The aggregate ID must be unique on the machine being configured.

Exporting the Episode File System

To export the Episode File System, replica the following steps:

a. Type **2** and press <**Return**>.

The **dce_config** script prompts:

```
Enter the device name for the aggregate to be exported (i.e. /dev/lvXX):
```

Note: The information in parentheses in the previous prompt is not the default value, but only an example of a value to be entered.

b. Type the pathname of the device on which the selected file system is stored and press **<Return>**.

The **dce_config** script prompts:

```
Enter the LFS fileset name (lfs_set):
```

c. Type the name of the LFS fileset and press **<Return>**.

The **dce_config** script prompts:

```
Do you want to format partition device_name as an Episode aggregate [n]?.
```

d. Type **y** to initialize the device named in a previous prompt as the aggregate to be exported as the Episode aggregate and press **<Return>**.

The **dce_config** script displays the following message:

S:****** *device_name* successfully initialized.

and then the following prompt:

```
Enter the LFS aggregate name (lfs_aggr):
```

e. Type a name for the Episode aggregate and press **<Return>**. The aggregate name must be unique on the machine being configured.

The **dce_config** script prompts:

```
Enter the LFS aggregate id (1):
```

f. Type a number for the aggregate ID and press **<Return>**. The aggregate ID must be unique on the machine being configured.

After you complete the steps listed in ''Exporting the Native File System'' or ''Exporting the Episode File System,'' the **dce_config** script exports the chosen file system and displays instructions on where to find information about exporting additional file systems by displaying:

```
S:****** Exporting device_name through DFS...
        If you wish to export additional aggregates, do so after
        completing this script by using the appropriate DFS
        administration commands described in the DFS Admin Guide.
        Press <RETURN> to continue, CTRL-C to exit:
```

7. Press <**Return**>.

   The **dce_config** script displays:

   ```
   S:****** Starting fxd...
   fx: FX server starts listening...
   ```

   and returns the Additional Server Configuration menu.

8.9.4.4 Configuring a DFS Fileset Location Database

Each cell that uses DFS requires at least one Fileset Location Database (FLDB) Server machine. Configure this machine immediately after configuring the DFS System Control machine.

To configure a Fileset Location Database machine, perform the following steps:

1. Type **6** on the Additional Server Configuration menu and press <**Return**>.

   The **dce_config** script displays the following message:

   ```
   S:****** Configuring DFS Fileset Location Database Server...
   ```

   and then prompts:

   ```
   Enter Cell Administrator's principal name: (cell_admin)
   ```

2. Type the name of the principal who was defined to be the initial privileged user of the registry database during the configuration of the master Security server and press <**Return**>.

   The **dce_config** script prompts:

```
Enter password:
```

3. Type the password for the initial privileged user's account and press **<Return>**. Note that this password is not displayed as you type it.

   The **dce_config** script loads the DFS kernel extensions and displays the following messages:

   ```
   S:****** Loading kernel extensions...
   rpc_config: installed krpc device at major number 71
   ```

   Then the **dce_config** script prompts for whether LFS (Episode) should be loaded

   ```
   Should the LFS Kernel Extension be loaded (n)?
   ```

4. Type **y** to configure the machine to use LFS or **n** to not and press **<Return>**.

   The **dce_config** script displays:

   ```
   Should LFS be initialized (n)?
   ```

5. Type **y** to initialize LFS or **n** to not and press **<Return>**.

   The **dce_config** script displays the following messages as it adds the appropriate DFS members to groups in the Security database.

```
S:****** Modifying the registry database for DFS server operation...

>>> group member added


>>> group member added

Current site is: registry server at /.../test_cell/subsys/dce/sec/master
Domain changed to: group
Current site is: registry server at /.../test_cell/subsys/dce/sec/master
Domain changed to: group
S:****** Starting bosserver...
Checking for a Ubik sync site in  hosts/node_name
Host /.:/hosts node_name/ is now the sync site
```

Then **dce_config** script prompts:

```
Enter the name of the system control machine:
```

6. Type the name of the machine configured as the system control machine and press <**Return**>. If you cell does not use a system control machine enter the name of the local machine.

   The **dce_config** script prompts:

   ```
   Enter the fileset name (root.dfs):
   ```

7. Type the name of the DFS file set and press <**Return**>.

   The **dce_config** script prompts:

   ```
           Enter the filesystem type for root.dfs:
           1. Native File System (e.g. UFS, JFS)
           2. Episode File System (LFS)
   selection:
   ```

   Your next steps depend on whether you want to export the native file system or the Episode file system. If you are exporting the native file system, go to the subsection titled ''Exporting the Native File System.'' If you are exporting the Episode file system, go to the subsection titled ''Exporting the Episode File System.''

   Exporting the Native File System

   To export the native file system, perform the following steps.

   a. Type **1** and press <**Return**>.

      The **dce_config** script prompts:

      ```
      Enter the aggregate name (/export):
      ```

      Note: The information in parentheses in the previous prompt is not the default value, but only an example of a value to be entered.

   b. Type the aggregate name and press <**Return**>.

You must specify the aggregate name as a mounted file system name, not as a physical device name. For example, to use the device **/dev/rz3c**, which is mounted as **/var**, specify **/var** as the aggregate name.

The **dce_config** script prompts:

```
Enter the aggregate ID (1):
```

c. Type the aggregate ID and press **<Return>**.

Exporting the Episode File System

To export the Episode File System, perform the following steps:

a. Type **2** and press **<Return>**.

The **dce_config** script prompts:

```
Enter the device name for the aggregate to be exported (i.e. /dev/lvXX):
```

Note: The information in parentheses in the previous prompt is not the default value, but only an example of a value to be entered.

b. Type the pathname of the device on which the selected file system is stored and press **<Return>**.

The **dce_config** script prompts:

```
Enter the LFS fileset name (lfs_set):
```

c. Type the name of the LFS fileset and press **<Return>**.

The **dce_config** script prompts:

```
Do you want to format partition device_name as an Episode aggregate [n]?.
```

d. Type **y** to initialize the device named in a previous prompt as the aggregate to be exported as the Episode aggregate and press **<Return>**.

The **dce_config** script displays the following message:

```
S:****** device_name successfully initialized.
```

and then the following prompt:

```
Enter the LFS aggregate name (lfs_aggr):
```

e. Type a name for the Episode aggregate and press <**Return**>. The aggregate name must be unique on the machine being configured.

The **dce_config** script prompts:

```
Enter the LFS aggregate id (1):
```

f. Type a number for the aggregate ID and press <**Return**>. The aggregate ID must be unique on the machine being configured.

After you complete the steps listed in ''Exporting the Native File System'' or ''Exporting the Episode File System,'' the **dce_config** displays information about the aggregate and about the servers it is starting:

```
number of sites: 1
   server           flags     aggr   siteAge principal      owner
host_name.ch.acme.   RW       1       0:00:00                <nil>
FLDB entry created for fileset root.dfs (0,,1) on aggregate 1 of node_name
S:****** Starting dfsbind...
S:****** Starting fxd...
fx: FX server starts listening...
```

and returns the Additional Server Configuration menu.

8.9.5 Configuring GDA Servers

The DCE Global Directory Agent (GDA) facilitates communication between DCE cells. If your cell will engage in intercell communication, the GDA server must be running. This section describes how to configure GDA servers.

You can configure a GDA server only on a previously configured client or CDS server machine.

You can configure GDA in a cell that uses either GDS or DNS as the global directory service. Although **dce_config** can configure GDA in a cell that uses GDS, you must use manual procedures to configure GDA in a cell that uses DNS. Both types of configuration are described in the subsections that follow.

8.9.5.1 Configuring GDA in a GDS Cell

To configure a GDA server in a cell that uses GDS for the global directory service, type **7** on the Additional Server Configuration menu and press **<Return>**.

The **dce_config** script displays the following messages, configures the GDA server and starts the GDA server daemon, **gdad**.

```
S:****** Configuring GDA Server...
S:****** Checking for active sec_client service...
S:****** Adding GDA principal to registry database...
S:****** Starting gdad...
```

When the configuration is complete, **dce_config** retuns the Additional Server Configuration menu.

8.9.5.2 Configuring GDA in a DNS Cell

To configure GDA in a DNS cell, perform the following steps:

1. Ensure that the machine being configured has access to the **named** daemon through **resolv.conf**.
2. Obtain the output of the **dcecp directory show/.:** command.
3. Register the cell information enumerated by the **dcecp directory show/.:** command with the **named** daemon (which is the server that controls DNS name resolution).
4. Restart the **named** daemon.

In a DNS cell, you can run the **gdad** daemon in the following modes:

DNS **gdad -x**

X.500 **gdad -b**

Both DNS and X.500
gdad

For information about starting and stopping **gdad**, and how to use the output of **dcecp directory show/.:**, see

8.9.6 Configuring Security Replicas

Security replicas help provide improved cell performance and reliability. To configure a security replica, perform the following steps:

1. At the Additional Server Configuration menu, type **8** and press <**Return**>.

   The **dce_config** script displays:

```
Enter the security replica name (without subsys/dce/sec) : (hostname)
```

2. Type the name to be assigned to the Security replica and press <**Return**>. Note that the default, *hostname* in parentheses, is replaced with the actual name of the host machine. The replica is created in the **subsys/dce/sec** directory.

   If this is the first time you have configured a Security replica on the machine, the **dce_config** script prompts:

```
What is the name of this cell (without /.../):
```

3. Type the name of the machine's cell and press <**Return**>.

   If the machine has not been configured as a DCE client machine, **dce_config** configures it as a DCE client. If it does this it displays the following message:

```
S:****** Configuring client...
```

   See the steps 1 through 4 in Section 8.9 for instructions on how to configure a client.

After you complete client configuration if necessary, the **dce_config** script displays the following messages as it creates and starts the replica on the local machine:

```
S******: Configuring Security Replication
Modifying acls on /.:/sec/replist ...
Modifying acls on /.:/subsys/dce/sec ...
Modifying acls on /.:/sec ...
Modifying acls on /.: ...
Modifying acls on /.:/cell-profile ...
```

Then, **dce_config** prompts for the replica's keyseed:

```
Enter keyseed for initial database master key:
```

4. Type in the text string for the keyseed, which is a temporary DES key that is used to generate the replica's master key (the key that the replica uses for account key creation). Press <**Return**>.

   The text you enter should not be easily guessed. Note that it is not displayed as you type it.

   The **dce_config** script creates and starts the security replica and displays:

   ```
   start slave security server (secd) ...
   ```

   The **dce_config** returns the Additional Server Configuration menu.

8.9.7 Configuring and Unconfiguring a Password Management Server

Password Management servers enable administrators to exert greater control over users' selection of passwords than that provided by DCE standard policy. You can create a password management server to enforce your cell's password-strength and password-generation requirements, or you can enhance the sample password management server provided with DCE.

For information on creating your own password management servers, see the chapter in the *OSF DCE Application Development Guide—Core Components* for more information about the administration of password management servers, see Chapter 30 in this guide.

To configure a password management server, type **10** at the Additional Server Configuration menu, and press **<Return>**. The **dce_config** script displays the following messages as it configures the password management server:

```
S:****** Configuring Password Management Server...
S:****** Successfully configured Password Management Server.
```

dce_config then returns to the Additional Server Configuration menu.

To unconfigure a password management server, type **11** at the Additional Server Configuration menu, and press **<Return>**. The **dce_config** script displays the following messages as it unconfigures the password management server:

```
S:****** Unconfiguring Password Management Server...
S:****** Successfully unconfigured Password Management Server.
```

dce_config then returns to the Additional Server Configuration menu.

8.9.7.1 Troubleshooting Password Management Configuration Errors

If attempts to configure or unconfigure a password management server fail, check for configuration error messages in the the password management server log file located in *dcelocal*/**var/security/pwd_strengthd.log**.

8.10 Configuring DCE Clients

After you configure your cell's master Security server and initial CDS server, you should configure each machine in the cell as a DCE client. Then you can configure additional servers and replicas as described in "Configuring Additional Servers."

When you configure a DCE client you set up the machine as a client of the core DCE Services: Security, CDS, and DTS.

To configure a DCE client machine, perform the following steps:

1. At the DCE Configuration Menu, type **3** and press <**Return**>.

   The **dce_config** script displays:

   ```
   S:****** Configuring client...
   ```

   The **dce_config** script prompts:

   ```
   What is the name of the Security Server for this cell:
   ```

2. Type the name of the machine that was configured as the master Security server and press <**Return**>.

   Then **dce_config** prompts for name of the initial CDS server machine:

   ```
   What is the name of the CDS server in this cell
   (if there is more than one, enter the name of
   the server to be cached if necessary) ?
   ```

3. Type the name of the machine that was configured as the initial CDS server and press <**Return**>.

   The **cdsadv** daemon finds the initial CDS server, and then prompts:

   Then, the **dce_config** script prompts:

   ```
   Create LAN profile so clients and servers can be
   divided into profile groups for higher performance in a multi-LAN
   cell? (n)
   ```

4. At this point you have two choices:

   a. Type **y** to configure the machine with multiple local area network (LAN) capabilities and press <**Return**>. You may want to use multiple LANs to organize how applications find objects in the namespace. When you respond

y, the **dce_config** script creates a LAN profile that allows the DTS clerks to synchronize with DTS servers on the local LAN.

The **dce_config** script prompts for the name of the LAN:

```
What is the name of the LAN?
```

Type the name and press <**Return**>. You can enter any arbitrary name. The name is used by **dce_config** to store cell profile information.

b. Type **n** to not configure the machine with multiple LAN capabilities and press <**Return**>.

The **dce_config** script displays the following message:

```
S:****** This node is now a CDS client
```

and then the following prompt:

```
Should this machine be configured as a DTS Clerk,
DTS Local Server, or DTS Global Server? (Default is DTS Clerk)
(clerk, local, global, none)
```

5. At this point you can type any of the following entries:
   - **clerk** to configure a DTS Clerk
   - **local** to configure a DTS Local server
   - **global** to configure a DTS Local server
   - **none** to not configure any DTS server on the machine

   If you choose to configure a DTS server, **dce_config** displays messages as it starts the appropriate DTS daemons. Then, **dce_config** completes the configuration and returns the DCE Configuration Menu.

8.11 Configuring DFS Clients

To configure a DFS client machine, perform the following steps:

1. At the DCE Configuration Menu, type **4** and press <**Return**>.

   The **dce_config** script prompts:

   ```
   Enter Cell Administrator's principal name: (cell_admin)
   ```

2. Type the name of the principal who was defined to be the initial privileged user of the registry database during the configuration of the master Security server and press <**Return**>.

   The **dce_config** script prompts:

   ```
   Enter password:
   ```

3. Type the password for the initial privileged user's account and press <**Return**>. Note that this password is not displayed as you type it.

   The **dce_config** script loads the DFS kernel extensions if any must be loaded and displays the following messages:

   ```
   S:****** Loading kernel extensions...
   rpc_config: installed krpc device at major number 71
   ```

   Then the **dce_config** script prompts for whether LFS (Episode) should be initialized:

   ```
   Should LFS be initialized (n)?
   ```

4. Type **y** to initialize LFS or **n** to not and press <**Return**>.

   The **dce_config** script displays:

   ```
           Is the cache :
           1. in memory
           2. on the local disk
   selection:
   ```

5. Type **1** if the cache is in memory or **2** if is is stored on the local disk and press <**Return**>.

The **dce_config** script starts the **dfsbind** daemon:

```
S:****** Starting dfsbind
```

and then prompts:

```
Enter the RAM size of the cache (10000):
```

6. Type the cache size and press **<Return>**.

   The **dce_config** script starts the **dfsd** daemon and returns the DCE Configuration Menu.

8.12 Configuring Auditing

If you wish to configure the DCE Auditing facility, type **9** and press **<Return>** at the DCE Configuration Menu.

The **dce_config** script displays:

```
S:****** Configuring Audit...
```

The **dce_config** script adds an entry for Auditing in the **/etc/rc.dce** file and starts the Audit daemon. It may also use the **dcecp** program to install default filters for auditing DTS, the Security Service, and the Auditing Service itself.

8.13 Building a Code Set Registry

A character set is a group of characters, such as the English alphabet, Japanese Kanji, and the European character set. A code set is a mapping of the members of a character set to specific numeric code values. Examples of code sets include ASCII, JIS X0208 (Japanese Kanji), and ISO 8859-1 (Latin 1). Different code set encodings exist for different character sets, but in addition, the same character set can be encoded in different ways.

A DCE cell automatically supports RPC applications that use the DCE Portable Character Set (DCE PCS), which is analogous to the U.S. English character set, and the ASCII and EBCDIC encodings for that character set. An internationalized DCE cell supports RPC applications that use non-English character sets and code sets other than ASCII and EBCDIC. In an internationalized DCE cell, multiple different character sets and code sets can exist, and also multiple different code sets for the same character set can exist. Operating systems generally use string names to refer to the code sets that the system supports. It is common for different operating systems to use different string names to refer to the same code set. For example, one system can use the name ISO8859-1 while another system can use Latin-1. Both names refer to the same code set.

The DCE code set registry provides a mechanism for uniquely identifying code sets and the character sets they encode across multiple heterogeneous operating systems in an internationalized DCE cell. The code set registry is a per-host file that contains mappings between the string names that the host's operating system platform uses for the code sets it supports and the unique identifiers for those code sets. Assigning a unique identifier to a code set provides internationalized DCE RPC clients and servers with a common representation to use when referring to a given code set.

If you are configuring an internationalized DCE cell, you need to build a code set registry on each machine in the cell. The next sections describe the steps involved.

8.13.1 Creating the Code Set Registry Source File

The Code Set Registry Compiler **csrc** creates a character and code set registry from the information supplied in a character and code set registry source file. Code set registry source files are created for input to **csrc** in two stages:

- During DCE licensee porting of DCE to one or more operating system platforms
- During the creation of an internationalized DCE cell or when a DCE machine is being configured for use in an internationalized DCE cell

In the first stage, DCE licensees create code set registry source files when they are porting DCE to a specific operating system platform and plan for their DCE product to support internationalized DCE applications. DCE licensees receive from OSF a template character and code set registry source file that contains the unique identifiers that OSF has assigned to the character sets and code sets that have been registered

with OSF. (This file exists in **src/rpc/csrc/csr/code_set_registry.txt**.) They modify this file to contain, for each code set that their platform supports, the local code set names for those supported code sets. They can also add to this file any vendor-specific, non-OSF registered code set names and values that their platform supports.

In the second stage, DCE cell administrators create code set registry source files when they are configuring machines that are part of an internationalized DCE cell. Cell administrators of internationalized DCE cells create their site-specific code set registry source files from one or more DCE licensee code set registry source files. Only one code set registry source file exists on each machine. The number of source files that need to be modified depends upon the number of DCE platforms that exist in the cell.

A code set registry source file is composed of a series of code set records. Each record describes, in human-readable form, the mapping between an OSF-registered, a licensee-defined, or a a site-specific unique code set value and the character string that a given operating system uses when referring to that code set. This character string is called the local code set name. Each code set record specifies one code set, and has the following form:

start
field_list
end

The *field_list* consists of the following keyword-value or keyword-text pairs:

description *text*
A comment string that briefly describes the code set.

loc_name *text*
A maximum 32-byte string (31 character data bytes plus a terminating NULL) that contains the operating system-specific name of a code set or the keyword **NONE**.

rgy_value *value*
A 32-bit hexadecimal value that uniquely identifies this code set. A registry value can be one that OSF has assigned or one that a DCE licensee or cell administrator has assigned.

char_values *value*[**:***value*]
One or more 16-bit hexadecimal values that uniquely identify each character set that this code set encodes. A character value can be one

that OSF has assigned or one that a DCE licensee or a cell administrator has assigned.

max_bytes *value*
A 16-bit value that specifies the maximum number of bytes this code set uses to encode one character.

Here is a sample of a licensee-supplied source file.

```
start
description ISO 8859-1:1987; Latin Alphabet No. 1
loc_name    iso88591
rgy_value   0x00010001
char_values 0x0011
max_bytes   1
end

start
description ISO 8859-2:1987; Latin Alphabet No. 2
loc_name    iso88592
rgy_value   0x00010002
char_values 0x0012
max_bytes   1
end

start
description ISO 8859-3:1988; Latin Alphabet No. 3
loc_name    iso88593
rgy_value   0x00010003
char_values 0x0013
max_bytes   1
end

start
description ISO 8859-4:1988; Latin Alphabet No. 4
loc_name    NONE
rgy_value   0x00010004
char_values 0x0014
max_bytes   1
end
```

For each different DCE platform that exists in the cell, the cell administrator takes that platform's licensee-generated character and code set registry source file and modifies the code set records within it to add the local code set names of any additional code sets that the site supports. (Note that the DCE licensees will have already modified the code set records for each code set that their DCE platform supports.) The cell administrator can also add to each platform-specific source file any site-specific, non-OSF registered code set names and values.

The cell administrator modifies the code set records that correspond to the code sets that the site supports as follows:

- For each code set that the site supports, replace the **NONE** keyword in the **loc_name** field of the code set record the name that your site uses to refer to the code set and the operating system code set converters associated with it. For example, in a UNIX environment, code set converters exist in the **iconv** directory. In this case, you would examine this directory to determine the names of the code set converters.
- Fill in the **description** field of the code set record to provide a detailed description of the code set and character set(s) that it supports. The text field can contain multiple lines; use the backslash character (\) to continue the line. If the site does not support a given code set, you must leave the **NONE** keyword in the code set record.
- Fill in the **max_byte** field of the code set record with the maximum number of bytes that the code set uses to encode one character. The count should include any single-shift control characters, if used.
- Add new values for any site-specific code set values or character set values that have not been registered with OSF to the appropriate **rgy_value** and **char_values fields**. These values must be in the range 0xf5000000 through 0xffffffff so that they do not collide with OSF-registered values. Use the colon character (:) to separate multiple character set values.

For additional source file usage information, see the **csrc(8dce)** reference page in the *OSF DCE Command Reference*.

8.13.2 Generating the Code Set Registry File

Cell administrators of internationalized DCE cells use the **csrc** utility to create site-specific code set registry files for each host in the cell. The cell administrator must run **csrc** once for each host that exists in the cell. You need to run **csrc** on a per-host basis because in some cases, the same operating system platform can be configured differently from host to host. For example, the same operating system platform can be multi-byte enabled on one machine, but single-byte enabled on another machine.

In order to run **csrc**, you need write permission to the **/usr/lib/nls/csr** directory, which generally requires **root** privilege.

When invoked without options, **csrc** uses the default source file **/usr/lib/nls/csr/code_set_registry.txt** and creates the default output file **/usr/lib/nls/csr/code_set_registry.db**. Use the **-i** and **-o** options to redirect **csrc** to use a specific source file or generate a specific output file. When **csrc** runs, it generates a log file named **CSRC_LOG** in the current (working) directory.

8.13.3 Adding Intermediate Code Sets

By default, the DCE RPC features that support internationalized RPC applications use the Universal code set ISO 10646 as the default intermediate code set (see the *OSF DCE Application Development Guide—Core Components* for an explanation of intermediate code sets and how they are used in internationalized RPC applications).

You can override this default by using the **-m** option to **csrc**, which adds a maximum of five intermediate code set names to the code set registry file's intermediate code set priority list. Specify the local code set name for the intermediate code set (not its code set identifier).

The order in which you specify intermediate code sets determines their order of precedence in the list; that is, the first intermediate code set you specify on the command line with **-m** becomes the first intermediate code set in the priority list, and thus will be the first code set used should an intermediate code set be required for internationalized RPC client-server communication.

8.13.4 Example

Here is a sample **csrc** command line:

```
csrc -i /test/i18n_app/code_set_registry.txt -o code_set_registry.db
-m euc -m sjis
```

In the previous example, the log file **CSRC_LOG** is created in the current directory, which is **/test/i18n_app**. The log file header shows that the EUC and SJIS code sets have been selected as intermediate code sets, and shows the priority scheme for using them:

```
Total CSRC Entry Count = 134
Effective CSRC Entry Count = 7

Intermediate Code Set Count = 2
Intermediate Code Set:
 Priority 1 = 1 Code Set Name = euc
 Priority 2 = 2 Code Set Name = sjis
```

Chapter 9

Managing DCE Configurations

This chapter describes how to use those **dce_config** functions that help you manage the installation and configuration process. These functions are as follows:

- **START**—To re-start DCE daemons
- **STOP**—To stop DCE daemons
- **UNCONFIGURE**—To remove entries for a configured client machine from the CDS namespace and from the Security database, essentially removing the client machine from the cell. Likewise to remove a CDS server, GDA server, or replica Security server from the cell.
- **REMOVE**—To stop DCE daemons and remove any data files created by those daemons

The **STOP** and **START** functions provide convenient ways to stop and stop all DCE daemons. The **UNCONFIGURE** and **REMOVE** functions allow you to return a machine to its state before it was configured, in effect reversing the effects of the configuration.

Note that during an install and during a configuration, you are prompted for whether or not to remove the remnants of a previous configuration. If you do, it has the same effects as running the **REMOVE** function described in this chapter.

The instructions given in this chapter assume that you have installed and are using the administrative utilities and other tools provided in DCE R1.1. The use of programs supplied in earlier versions of DCE can cause problems or prevent you from doing what you want to do.

The remainder of this chapter takes you through the steps to use the **START**, **STOP**, **UNCONFIGURE** and **REMOVE** functions. All the functions are chosen from the DCE Main Menu, described in Chapter 6. With the exception of the **UNCONFIGURE** function, which can be run on a remote machine, all functions described in the chapter must be run on the local machine.

9.1 Starting DCE Daemons

As part of configuring a machine, the **dce_config** script starts all configured daemons. The **START** function provides a convenient way of restarting all DCE daemons that have been successfully configured.

The **START** function invokes the **/etc/rc.dce** component script and, if DFS is installed on the machine, the **/etc/rc.dfs** component script. You can invoke these scripts directly. (The component scripts are described in Chapter 10.) In addition, you can start any DCE daemon manually using standard commands.

To use the **START** function to start all DCE daemons, at the DCE main Menu, type **3** and press **<Return>**.

The **dce_config** script attempts to start all daemons that have been configured on the machine. It displays what it is doing as it does it. When the daemons are started, **dce_config** returns the DCE Main Menu.

The following is a sample of the information displayed by **dce_config** after you select the **START** function on a server machine; for a client machine, the display is shorter because fewer daemons are started. Note that when you run this function, it may start some platform-specific daemons not shown in the sample display.

```
S:****** Attempting to start all configured DCE daemons...
        /opt/dcelocal/bin/dced
        /opt/dcelocal/bin/secd
        /opt/dcelocal/bin/cdsadv
        /opt/dcelocal/bin/cdsd
        /opt/dcelocal/bin/dtsd
        /opt/dcelocal/bin/dts_null_provider -p 60 -i 100
S:****** Successfully started all configured DCE daemons.
```

9.2 Stopping DCE Daemons

The **dce_config STOP** function provides a convenient way of stopping all DCE daemons running on the local machine.

The **STOP** function invokes the **/etc/dce_shutdown** component script and, if DFS is installed on the machine, the **/etc/dfs.clean** component script. You can invoke these scripts directly. (The component scripts are described in Chapter 10.) In addition, you can stop any DCE daemon manually using standard commands.

To stop all DCE daemons running on the machine, at the DCE main Menu, type **4** and press <**Return**>.

The **dce_config** script attempts to stop all daemons that are running on the machine and displays the following message:

```
S:****** Attempting to stop all running DCE daemons...
```

After **dce_config** stops the daemons, it displays the following message and returns the DCE Main Menu.

```
S:****** Successfully stopped all running DCE daemons.
```

9.3 Unconfiguring Client and Server Machines

The **UNCONFIGURE** function is used to reverse the effects of configuring a client or server machine. This function unconfigures machines by removing their entries from the cell namespace and Security registry.

The **UNCONFIGURE** function only unconfigures CDS servers and Security replica servers. It does not unconfigure the Security master server.

To completely unconfigure a machine, you must also run the **REMOVE** function. The **REMOVE** function will delete any files created during the machine's configuration.

Before you can run the **UNCONFIGURE** function, all the DCE daemons on the machine must be running. If you need to restart daemons, run the **START** function.

If the configuration of a Security and CDS client did not complete successfully, you must unconfigure the machine from some other machine in the cell.

The **UNCONFIGURE** function invokes the **/etc/dce.unconfig** component script and, if DFS is installed on the machine, the **/etc/dfs.unconfig** component script. You can invoke these component scripts directly. (The component scripts are described in Chapter 10.)

Unlike other **dce_config** functions, you can run the **UNCONFIGURE** function remotely.

The steps to unconfigure either a server or client machine are the following:

1. At the DCE Main Menu, type **5** and press **<Return>**.

   The **dce_config** script displays the following message:

   ```
   S:****** Attempting to unconfigure a node from the
   cell name space...
   ```

   Then, **dce_config** prompts for the name of the machine to unconfigure:

   ```
   Enter hostname of node to be unconfigured: (host_name)
   ```

2. Type the name of the machine to be unconfigured and press <**Return**>.

   The **dce_config** script displays the following prompt:

   ```
   Unconfiguring a node will remove the node's ability
   to operate in the cell.  A reconfiguration of the node
   will be required.
   Do you wish to continue (y/n)? (n)
   ```

3. Type **y** to continue and press <**Return**>.

   If you continue, **dce_config** prompts:

   ```
   Enter the Cell Administrator's principal name: (cell_admin)
   ```

   Type the name of the principal who was defined to be the initial privileged user of the registry database during the configuration of the master Security server and press <**Return**>.

   The **dce_config** script prompts:

   ```
   Enter password:
   ```

4. Type the password for the initial privileged user's account and press <**Return**>. Note that this password is not displayed as you type it.

   The **dce_config** script deletes the entries in the registry and the CDS namespace for the machine.

   When it completes the unconfigure, **dce_config** displays the DCE Main Menu.

Note if you are unconfigure a host on that host, the **dce_config** script will automatically run the **dce_shutdown** component script to shutdown all the DCE daemons.

9.4 Removing the Results of a Configuration

The **REMOVE** function removes the results of a client or server machine configuration. Before using **REMOVE** function on a machine, run the **UNCONFIGURE** function on that machine.

Remember that each cell requires a master Security server and a CDS server. If you remove these servers in the process of unconfiguring either a client or server machine, you must reconfigure them and all client machines for the cell to operate.

The **REMOVE** function invokes the **/etc/dce_shutdown** component script. If the any DCE daemons are running when you run the **REMOVE** function, the **/etc/dce_shutdown** component, will invoke the **/etc/dce.clean** component script to stop them. Similarly, if DFS is installed on the machine, the **REMOVE** function invokes the **/etc/dfs.rm** component script and, if necessary, the **/etc/dfs.clean** component script.

You can invoke these component scripts directly. If you do, be sure to invoke the **.shutdown** and **.clean** scripts before the **.rm** scripts. (The component scripts are described in Chapter 10.)

To remove the results of a machine configuration, perform the following steps.

1. At the DCE Main Menu, type **6** and press **<Return>**.

   The **dce_config** script prompts to remind you that the **REMOVE** function removes all effects of the last configuration that client machines must be unconfigured before **REMOVE** is run.

```
REMOVE will remove the node's ability to operate in the cell.
A reconfiguration of the node will be required.  If this is not
a server node, then this node should be unconfigured before a
REMOVE is done.  Do you wish to continue (y/n)? (n)
```

2. Press **<Return>** to continue.

   The **dce_config** script proceeds to remove all effects of the configuration of all DCE components on the machine. As it does, it displays the following messages.

```
S:****** Attempting to stop all running DCE daemons...
S:****** Successfully stopped all running DCE daemons.
```

```
S:****** Attempting to remove all remnants of previous DCE configurations...
S:****** Successfully removed all remnants of previous DCE configurations
         for all components.
```

When the process is complete, **dce_config** displays the DCE Main Menu.

Note: The **REMOVE** function does not delete the binaries installed on the machine during installation. To do this, invoke the **dce.rm** component script with the **install** option for the core component binaries or the **dfs.rm** component script with the **install** option for the DFS binaries.

Chapter 10

Customizing the dce_config Processing

This chapter contains information useful for customizing **dce_config** processing. Specifically, it describes:

- Automating **dce_config** processing
- Setting the **dce_config** environment variables
- Controlling the logging of messages
- Using the the **dce_config** component scripts

10.1 Automating dce_config Processing

Using an environment file and a command file, you can automate **dce_config** processing.

The environment file sets the DCE and DFS environment variables. These variables provide answers to the user prompts encountered during interactive processing. This means that you do not have to answer the prompts, instead **dce_config** obtains the

answers from the variable settings. (Note that you can also set the variables as shell variables. Sections 10.2.1 and 10.2.2 describe the variables and their settings.)

The command file is a shell script that initiates installation and configuration processing. If you use a command file, **dce_config** does not display the menus and initiate interactive processing, but instead sources the command file for instructions.

If you use both the environment and command file, you can completely automate **dce_config** processing. If you use only the environment file, you can partially automate **dce_config** processing. In this case although you must make the menu selections that indicate which components to install and configure, you are not required to answer the prompts displayed during a non-automated installation and configuration.

10.1.1 Using the Environment and Command Files

To use the files, invoke **dce_config** as follows:

dce_config -e *environment_file* **-c** *command_file*

The **-e** option sources the named environment file at **dce_config** startup. The **-c** option sources the named command file at **dce_config** startup.

10.1.2 Sample Environment File

A sample environment file, **config.env**, is provided by OSF with the DCE source. You can copy the file and use it as supplied or you can use it as guide to creating your own environment file. The sample file is not copied to the install tree during DCE installation. Figure 10-1 shows this file.

The file entries are in the form:

variable=value

To change a value, simply replace it with the new value. The possible values are listed as comments (prefaced with #) just following the variable.

Figure 10–1. Sample Environment File

```
# This file contains most of the variable used by dce_config.
# You can pass this on the command line via the -e switch
# or dce_config will look at /etc/dce_config.conf by default.
#
# This is a shell script sourced in to dce_config at startup
# and can contain shell commands if you wish.
#

#
# General config values
#

EXIT_ON_ERROR=n         # y/n Exit if fatal error encountered, default n
DO_CHECKS=n             # y/n Prompt for continue after warnings, default y

CELL_ADMIN=cell_admin   # Principal name of Cell Administrator
CELL_ADMIN_PW=-dce-     # Password for CELL_ADMIN account

#TOLERANCE_SEC=120      # Number of seconds clients can differ from sec server
check_time=y            # y/n if you want to check times, default y
SYNC_CLOCKS=y           # y/n Sync our clock to $TIME_SERVER
TIME_SERVER="<host>"    # Any host which has dts server on it,
                        # usually the security server ($SEC_SERVER)


#
# Install
#
REMOVE_PREV_INSTALL=y           # y/n Remove previous install before
                                # installing anything.
mach=‘uname‘
case $mach in
        OSF1)   machine=at386;;
        AIX)    machine=rios;;
        HP-UX)  machine=hp800;;
esac
# path to install area
#FILESYSTEM="install/${machine}/opt/dce1.1"
```

```
#MEDIA="<device>"                       # device to read tape from

CP_OR_SYMLINK=copy          # Use copy or sym-links(link) to install
USE_DEF_MSG_PATH=y          # y/n use default message catalog path
USE_ETHER_FILE=y            # y/n Trust /etc/ieee_802_addr file (OSF/1)
DO_LIB_ADMIN=y              # y/n Install libdce.so on OSF/1 system

#
# DFS install
#
INSTALL_OPT_SERS=y              # y/n install optional servers (bak, butc, etc)
INSTALL_OPT_CLIENT=y            # y/n install optional client binaries (cm, etc)


#
# General config variables
#
REMOVE_PREV_CONFIG=y    # y/n Remove previous config before config-ing anything

CELL_NAME="<cellname>"  # Name of cell to configure

#HOST_NAME_IP="<ip_address>"    # IP address of current host (if getip fails)


#
# DCE Client config
#
DTS_CONFIG=clerk                # When doing DCE client config, what should
                                # we be? clerk, local, global, none

#
# Security Config
#
SEC_SERVER="<host>"             # Hostname of security server
#SEC_SERVER_IP="<ip-address>"   # fallback if getip program doesn't work
KEYSEED="garBageMan"            # Keyseed for initial database master

#
# Default values are provided, for PWD_MGMT_SVR and PWD_MGMT_SVR_OPTIONS.
#
```

```
# PWD_MGMT_SVR="$DCELOCAL/bin/pwd_strengthd" # Path to Password Mgmt Server
# PWD_MGMT_SVR_OPTIONS="-v"     # Options for Password Management Server

#SEC_REPLICA="<hostname>"       # Name of the security replica database


#
# CDS Config
#
CACHE_CDS_SERVER="<hostname>"   # Name of a cds server to cache
#CACHE_CDS_SERVER_IP="<ip_address>" # fallback if getip program doesn't work
MULTIPLE_LAN=n                  # y/n do you have multiple lans
#LAN_NAME="<name>"              # Name of lan if MULTIPLE_LAN=y

#REP_CLEARINGHOUSE="<name_ch>"  # Name for new replica clearing house
#DIR_REPLICATE="n"              # y/n manually type in more directories
                                # to replicate.

#
# GDA
#

#
# DTS Config
#
NTP_HOST="<hostname>"           # Name of ntp server


#SEC_REPLICA="<hostname>"       # Name of the security replica database


#
# CDS Config
#
CACHE_CDS_SERVER="<hostname>"   # Name of a cds server to cache
#CACHE_CDS_SERVER_IP="<ip_address>" # fallback if getip program doesn't work
MULTIPLE_LAN=n                  # y/n do you have multiple lans
#LAN_NAME="<name>"              # Name of lan if MULTIPLE_LAN=y
```

```
#REP_CLEARINGHOUSE="<name_ch>"  # Name for new replica clearing house
#DIR_REPLICATE="n"              # y/n manually type in more directories
                                # to replicate.

#
# GDA
#

#
# DTS Config
#
NTP_HOST="<hostname>"           # Name of ntp server

#
# DFS Config
#

AGG_FS_TYPE=native            # native/episode aggregate fs type to export
AGG_DEV_NAME="<device>"    # device name for the aggregate to be exported
AGG_MOUNT_PATH="<path>"    # mount path for aggregate
AGG_NAME="<name>"          # Name of aggregate
AGG_ID="<number>"          # numerical id of aggregate

CACHE_SIZE_RAM=10000          # number of bytes for memory cache
CACHE_SIZE_DISK=10000         # number of bytes for disk cache
CACHE_DIR_DISK="/opt/dcelocal/var/adm/dfs/cache" # pathname of disk cache
CLIENT_CACHE_LOC=disk         # mem/disk where is the cache
CONFIG_NFS_GATEWAY=n          # configure dfs client as nfs gateway

EPI_FORMAT_PART=n             # y/n format partition as episode
EPI_FORCE_INIT=n              # y/n force initialization, even if data loss
INIT_LFS=n                    # y/n initialize the LFS (using epiinit)?
LOAD_LFS_KEXT=n               # y/n load LFS kernel extensions

ROOT_FILESET_NM="<name>"        # Root fileset name
SCM_NAME="<hostname>"           # System Control Machine name


# This file contains most of the variable used by dce_config.
# You can pass this on the command line via the -e switch
```

```
# or dce_config will look at /etc/dce_config.conf by default.
#
# This is a shell script sourced in to dce_config at startup
# and can contain shell commands if you wish.
#

#
# General config values
#

EXIT_ON_ERROR=n         # y/n Exit if fatal error encountered, default n
DO_CHECKS=n             # y/n Prompt for continue after warnings, default y

CELL_ADMIN=cell_admin   # Principal name of Cell Administrator
CELL_ADMIN_PW=-dce-     # Password for CELL_ADMIN account

#TOLERANCE_SEC=120      # Number of seconds clients can differ from sec server
check_time=y            # y/n if you want to check times, default y
SYNC_CLOCKS=y           # y/n Sync our clock to $TIME_SERVER
TIME_SERVER="<host>"    # Any host which has dts server on it,
                        # usually the security server ($SEC_SERVER)


#
# Install
#
REMOVE_PREV_INSTALL=y           # y/n Remove previous install before
                                # installing anything.
mach=‘uname‘
case $mach in
        OSF1)   machine=at386;;
        AIX)    machine=rios;;
        HP-UX)  machine=hp800;;
esac
# path to install area
#FILESYSTEM="install/${machine}/opt/dce1.1"
#MEDIA="<device>"                       # device to read tape from

CP_OR_SYMLINK=copy              # Use copy or sym-links(link) to install
USE_DEF_MSG_PATH=y              # y/n use default message catalog path
```

```
USE_ETHER_FILE=y                # y/n Trust /etc/ieee_802_addr file (OSF/1)
DO_LIB_ADMIN=y                  # y/n Install libdce.so on OSF/1 system

#
# DFS install
#
INSTALL_OPT_SERS=y              # y/n install optional servers (bak, butc, etc)
INSTALL_OPT_CLIENT=y            # y/n install optional client binaries (cm, etc)


#
# General config variables
#
REMOVE_PREV_CONFIG=y    # y/n Remove previous config before config-ing anything

CELL_NAME="<cellname>"  # Name of cell to configure

#HOST_NAME_IP="<ip_address>"    # IP address of current host (if getip fails)


#
# DCE Client config
#
DTS_CONFIG=clerk                # When doing DCE client config, what should
                                # we be? clerk, local, global, none

#
# Security Config
#
SEC_SERVER="<host>"             # Hostname of security server
#SEC_SERVER_IP="<ip-address>"   # fallback if getip program doesn't work
KEYSEED="garBageMan"            # Keyseed for initial database master

#
# Default values are provided, for PWD_MGMT_SVR and PWD_MGMT_SVR_OPTIONS.
#

# PWD_MGMT_SVR="$DCELOCAL/bin/pwd_strengthd" # Path to Password Management Serve
r
# PWD_MGMT_SVR_OPTIONS="-v"     # Options for Password Management Server
```

```
#SEC_REPLICA="<hostname>"       # Name of the security replica database


#
# CDS Config
#
CACHE_CDS_SERVER="<hostname>"   # Name of a cds server to cache
#CACHE_CDS_SERVER_IP="<ip_address>" # fallback if getip program doesn't work
MULTIPLE_LAN=n                  # y/n do you have multiple lans
#LAN_NAME="<name>"              # Name of lan if MULTIPLE_LAN=y

#REP_CLEARINGHOUSE="<name_ch>"  # Name for new replica clearing house
#DIR_REPLICATE="n"              # y/n manually type in more directories
                                # to replicate.

#
# GDA
#

#
# DTS Config
#
NTP_HOST="<hostname>"           # Name of ntp server


#SEC_REPLICA="<hostname>"       # Name of the security replica database


#
# CDS Config
#
CACHE_CDS_SERVER="<hostname>"   # Name of a cds server to cache
#CACHE_CDS_SERVER_IP="<ip_address>" # fallback if getip program doesn't work
MULTIPLE_LAN=n                  # y/n do you have multiple lans
#LAN_NAME="<name>"              # Name of lan if MULTIPLE_LAN=y

#REP_CLEARINGHOUSE="<name_ch>"  # Name for new replica clearing house
#DIR_REPLICATE="n"              # y/n manually type in more directories
                                # to replicate.
```

```
#
# GDA
#

#
# DTS Config
#
NTP_HOST="<hostname>"           # Name of ntp server

#
# DFS Config
#

AGG_FS_TYPE=native              # native/episode aggregate fs type to export
AGG_DEV_NAME="<device>"         # device name for the aggregate to be exported
AGG_MOUNT_PATH="<path>"         # mount path for aggregate
AGG_NAME="<name>"                       # Name of aggregate
AGG_ID="<number>"                       # numerical id of aggregate

CACHE_SIZE_RAM=10000            # number of bytes for memory cache
CACHE_SIZE_DISK=10000           # number of bytes for disk cache
CACHE_DIR_DISK="/opt/dcelocal/var/adm/dfs/cache" # pathname of disk cache
CLIENT_CACHE_LOC=disk           # mem/disk where is the cache
CONFIG_NFS_GATEWAY=n            # configure dfs client as nfs gateway

EPI_FORMAT_PART=n               # y/n format partition as episode
EPI_FORCE_INIT=n                # y/n force initialization, even if data loss
INIT_LFS=n                      # y/n initialize the LFS (using epiinit)?
LOAD_LFS_KEXT=n                 # y/n load LFS kernel extensions

ROOT_FILESET_NM="<name>"        # Root fileset name
SCM_NAME="<hostname>"           # System Control Machine name
```

10.1.3 Sample Command File

A sample command file, **config.cmd**, is provided by OSF with the DCE source. You can copy the file and use it as supplied or you can use it as guide to creating your own environment file. The sample file is not copied to the install tree during DCE installation. Figure 10-2 shows this file.

The file consists of **install** and **config** command lines and comment lines that document the script's actions. The **install** lines specify the component to install and, for DFS, the type of server (System Control Machine, Private File Server, File Server, or File Location Database Server). The **config** lines specify:

- The component to configure
- Whether to configure the component as a client, server, or replica (for Security and GDS)
- Whether to configure the component as a local server, global server, clerk, or time provider (for DTS)
- Whether to configure the component as a System Control Machine, Private File Server, File Server, or File Location Database Server (for DFS)

The file is thoroughly annotated and can be used simply by uncommenting the lines that install and configure the components you want. The # character indicates a comment line. Remove the # to uncomment the line. For example, to install the Security server, change the following line:

```
#install sec   # Security Server
```

to look like:

```
install sec   # Security Server
```

Note that the text `# Security Server` following `install sec` is a comment that documents what will be installed by the line. Note also that comments also appear at the beginning of each logical grouping of actions. For example, the following comment appears at the beginning of the lines that install components:

```
# install commands
#
#-----------------------------------------------------
```

```
#
#  install <what>
#  <what> :=            sec     gds     appdev          sec-replica
#                       cds     dts     cdsbrowser
#                       client  nidl_to_idl
#
#  install dfs  <which>
#               <which> := client|scm|privatefs|fs|fldb
```

Figure 10–2. Sample Command File

```
#
# This file is an example of what you can pass to dce_config
# via the -c switch.  It it sourced in to dce_config, and
# can contain shell script commands if you wish.

#
# install commands
#
#-----------------------------------------------------
#
#  install <what>
#  <what> :=            sec     appdev          sec-replica
#                       cds     dts     cdsbrowser
#                       client  nidl_to_idl
#
#  install gds  <which>
#               <which> := client|server
#
#  install dfs  <which>
#               <which> := client|scm|privatefs|fs|fldb
#

#install sec                    # Security Server
#install cds                    # CDS Server
#install dts                    # DTS Server
#install client                 # DCE Client
#install appdev                 # Application Development Environment
#install sec-replica            # Replica Security Server
#install cdsbrowser             # Install cdsbrowser
```

```
#install nidl_to_idl            # Install nidl_to_idl

#
# GDS install
#
#install gds client
#
# You do not need to install gds client if you install the gds server.
#
#install gds server

# DFS install
#
#install dfs client
#
# You do not need to install dfs client if you install one of the servers.
#
#install dfs scm
#install dfs privatefs
#install dfs fs
#install dfs fldb


#
# config commands
#
#-----------------------------------------------------
#  config  <component> <how>
#
#  <component> :=
#              client
#              sec     <how> :=        client|server|replica
#              cds     <how> :=        client|server|replica
#              gda
#
#              dts     <how> :=        clerk|local|global|server|none
#                                      ntp-provider|null-provider
#
#              dfs     <how> :=        client|scm|privatefs|fs|fldb
#
```

```
#config client                  # Same as:
                                #  config sec client
                                #  config cds client
                                #  config dts $DTS_CONFIG
#
# Security
#
# Can only pick one, server implies client.
#config sec client              # Security Client
#config sec server              # Security Server
#config sec replica             # Security Replica


#
# CDS
#
# Can only pick one, server implies client.
#config cds client              # CDS Client
#config cds server              # CDS Server
#config cds replica             # Additional CDS server on this machine

#
# GDA
#
#config gda                     # Run a gdad on this machine

#
# Audit subsystem
#
#config audit                   # Fire up auditd

#
# DTS
#
# Can only pick one, server implies client.
#config dts clerk               # DTS Clerk
#config dts local               # DTS Local Server
#config dts global              # DTS Global Server

#
```

```
# DTS Time providers
#
# Can only pick one
#config dts ntp-provider        # Run NTP provider on this node
#config dts null-provider       # Run Null provider on this node

#
# DFS
#
# You may pick one of these three server types
#config dfs fldb                # file Location Database server
#config dfs fs                  # File Server
#config dfs privatefs           # Private File Server

# Any of the above can be a SCM.
#config dfs scm                 # System Control Machine

#
# Client must be configured after server
#config dfs client              # DFS Client


#
# This file is an example of what you can pass to dce_config
# via the -c switch.  It it sourced in to dce_config, and
# can contain shell script commands if you wish.

#
# install commands
#
#-----------------------------------------------------
#
#  install <what>
#  <what> :=            sec     appdev          sec-replica
#                       cds     dts     cdsbrowser
#                       client  nidl_to_idl
#
#  install gds  <which>
#               <which> := client|server
#
```

```
#  install dfs  <which>
#               <which> := client|scm|privatefs|fs|fldb
#

#install sec                    # Security Server
#install cds                    # CDS Server
#install dts                    # DTS Server
#install client                 # DCE Client
#install appdev                 # Application Development Environment
#install sec-replica            # Replica Security Server
#install cdsbrowser             # Install cdsbrowser
#install nidl_to_idl            # Install nidl_to_idl

#
# GDS install
#
#install gds client
#
# You do not need to install gds client if you install the gds server.
#
#install gds server

# DFS install
#
#install dfs client
#
# You do not need to install dfs client if you install one of the servers.
#
#install dfs scm
#install dfs privatefs
#install dfs fs
#install dfs fldb


#
# config commands
#
#-----------------------------------------------------
#  config  <component> <how>
#
```

```
#  <component> :=
#              client
#              sec     <how> :=        client|server|replica
#              cds     <how> :=        client|server|replica
#              gda
#
#              dts     <how> :=        clerk|local|global|server|none
#                                      ntp-provider|null-provider
#
#              dfs     <how> :=        client|scm|privatefs|fs|fldb
#
#config client                 # Same as:
                               #  config sec client
                               #  config cds client
                               #  config dts $DTS_CONFIG
#
# Security
#
# Can only pick one, server implies client.
#config sec client             # Security Client
#config sec server             # Security Server
#config sec replica            # Security Replica


#
# CDS
#
# Can only pick one, server implies client.
#config cds client             # CDS Client
#config cds server             # CDS Server
#config cds replica            # Additional CDS server on this machine

#
# GDA
#
#config gda                    # Run a gdad on this machine

#
# Audit subsystem
#
```

```
#config audit                   # Fire up auditd

#
# DTS
#
# Can only pick one, server implies client.
#config dts clerk               # DTS Clerk
#config dts local               # DTS Local Server
#config dts global              # DTS Global Server

#
# DTS Time providers
#
# Can only pick one
#config dts ntp-provider        # Run NTP provider on this node
#config dts null-provider       # Run Null provider on this node

#
# DFS
#
# You may pick one of these three server types
#config dfs fldb                # file Location Database server
#config dfs fs                  # File Server
#config dfs privatefs           # Private File Server

# Any of the above can be a SCM.
#config dfs scm                 # System Control Machine

#
# Client must be configured after server
#config dfs client              # DFS Client
```

10.2 Setting Environment Variables

The **dce_config** script and the **dfs_config** component script recognize and use a number of environment variables. You can use these variables to supply information to **dce_config** instead of typing the information in in response to prompts.

You can set the variables in any of the three following ways:

- As shell environment variables
- In an environment file that you create for use with the **-e** option of the **dce_config** script
- In a file named **dce_config.conf** that you create (in the **/etc** directory) for use by the **dce_config** script

If you invoke **dce_config** with the **-e** option and provide the name of the environment file, **dce_config** sources this file. If you invoke **dce_config** without the **-e** option, **dce_config** tries to source the file named **dce_config.conf**. If the file does not exist, it uses your shell variable settings to provide answers to the prompts. If you have not set your shell variables, **dce_config** prompts for information it needs.

The **dce_com_env** file sets the internal variables for the **dce_config** and for the **dfs_config** component script. The **dce_config_env** file describes the variables you can set, Note that the log file produced by the message logging facility also lists the current settings of the environment variables.

The following subsections describe the **dce_config** and **dfs_config** variables.

10.2.1 The dce_config Environment Variables

Table 10-1 lists the environment variable names and their values. In the table, the term default refers to the setting assigned to the environment variable by OSF.

Table 10–1. dce_config Environment Variables

Variable	Value
CACHE_CDS_SERVER	The name of the CDS server to cache. It is not required that the cached server be the initial CDS Server. Used during CDS client configuration.
CACHE_CDS_SERVER_IP	The IP address of the CDS server to cache.

Variable	Value
CELL_ADMIN	The principal name of the initial privileged user of the registry database (known as the "registry creator"). Used during Security server configuration.
CELL_ADMIN_PW	The default password assigned to the accounts created when the registry database is created, including the account for the registry creator. The default is **-dce-**.
CELL_NAME	The name of the cell (without the **.../**) on which the configuration is being performed. Used during Security server configuration.
CHANGE_PW	Indicates whether or not **dce_config** displays `Password must be changed` on exiting when the cell administrator password (CELL_ADMIN_PW) is the same as the default password. The default is **n**. It is recommended that you do not change this value in order to help ensure that the cell administrator is not assigned a commonly known password. This variable is used in conjunction with the DEFAULT_PW variable.

Variable	Value
CHECK_TIME	Specifies whether or not to check client and server clock synchronization: **y** indicates the time will be checked; **n** indicates it will not. The default is **y**. If you execute **dce_config** from a ''here'' file, set CHECK_TIME to **n** since time checking uses a **telnet** command that causes input from the "here" file to be lost. Note that **dce_config** do not recognize time zones. If you are configuring a cell across time zones, set CHECK_TIME to **n**.
DC_DISPLAY_THRESHOLD	Specifies the messages to write to stdout. Possible values are **ERROR**, **WARNING**, **SUMMARY**, **DETAIL**, **VERBOSE**, and **DEBUG**. The default is SUMMARY.
DC_LOG_THRESHOLD	Specifies the Minimum priority log messages to write to the log file, **/tmp/dce_config.log**. Possible values are **ERROR**, **WARNING**, **SUMMARY**, **DETAIL**, **VERBOSE**, and **DEBUG**. The default is DEBUG.
DEFAULT_MAX_ID	The highest value UNIIX ID for principals. The default value is 32767, which means that only principals with UNIX IDs lower than 32767 can access the cell. It is recommended that you accept the default. Used during Security Server configuration.

Variable	Value
DEFAULT_PW	Contains the default password used when the registry is created. This variable is used to determine if the cell administrator's password (CELL_ADMIN_PW) is the same as the default password. When the user exits **dce_config**, the value of DEFAULT_PW and CELL_ADMIN_PW are checked. If they are the same and if the CHANGE_PW variable is set **Y**, **dce_config** issues the warning message `Password must be changed`. The default for this variable is **-dce-**. If your site has a commonly used and known password, change the DEFAULT_PW variable to that password to help ensure that the cell administrator account is not assigned a commonly known password.
DIR_REPLICATE	Controls the replication of CDS directories when an additional CDS server is being created at DCE configuration time. The value **y** will cause **dce_config** to prompt for more directories to replicate; **n** will not. The default is **n**.

Variable	Value
DO_CHECKS	Controls the display of three prompts. The first is whether or not the `Press <RETURN> to continue, CTRL-C to exit:` prompt is returned when **dce_config** encounters a non-fatal error. This prompt forces the user to acknowledge the error and offers a way to exit **dce_config**. The second and third prompt occur during master Security server configuration. They prompt for a UNIX ID number at which the Security server will start assigning automatically generated group UNIX IDs and principal UNIX IDs. If this prompt is turned off, the default is the default described in the **DEFAULT_MAX_ID** and **GID_GAP** variables. For the **DO_CHECKS** variable, **y** displays the prompt; **n** does not. The default is **y**.
EXIT_ON_ERROR	An indication of whether or not **dce_config** will exit in the event of a fatal error: **y** indicates that **dce_config** exits when it encounters a fatal error; **n** indicates it will not. The default is **n**. Setting this variable to Y OR N can help prevent a "here" file from getting out of sync with **dce_config**.
GID_GAP	The increment above highest currently used GID at which the Security service will start assigning automatically generated GIDs. The value of this variable is used with the LOW_GID variable to set the starting point for UIDs automatically assigned by the Security server. Default is 100. Used in Security server configuration.

Variable	Value
HOST_NAME_IP	The IP address of node on which **dce_config** is running.
KEYSEED	A character string used to seed the random key generator in order to create the master key for the master and each slave database. Each database has its own master key and thus keyseed. Used in Security server configuration.
LAN_NAME	For multiple LAN configurations, the internal name of the LAN (in the LAN profile). Used in CDS server configuration.
LOGFILE	The full pathname of the log file produced by **dce_config**. The default is **/tmp/dce_config.log**
LOW_GID	The value at which the Security server will start assigning automatically generated group IDs. The default is the value of the highest group ID currently used on the machine being configured, incremented by the value of GID_GAP. Although there is no restriction that the value of LOW_GID must be higher than the machine's highest group ID, if you supply a LOW_GID that is less than or equal to the highest currently used group ID, **dce_config** issues a warning message and prompts the user to reenter LOW_GID. Used in master Security server configuration.

Variable	Value
LOW_UID	The value at which the Security Server will start assigning automatically generated UNIX IDs. The default is the value of the highest UNIX ID currently used on the machine being configured, incremented by the value of UID_GAP. Although there is no restriction that the value of LOW_UID must be higher than the machine's highest UNIX ID, if you supply a LOW_UID that is less than or equal to the highest currently used UNIX ID, **dce_config** issues a warning message and prompts the user to reenter LOW_UID. Used in master Security server configuration.
MULTIPLE_LAN	An indication of whether or not to configure the node with multiple LAN capabilities: **y** indicates configure with multiple LAN capabilities, **n** indicates do not. Used in CDS configuration.
NTP_HOST	The name of the host on which the NTP time provider server is running. Used in DTS Time Provider configuration.
PWD_MGMT_SVR	The default pathname to the Password Management server, which is **$DCELOCAL/bin/pwd_strength**. Used in Password Management server configuration.
PWD_MGMT_SVR_OPTIONS	The default options for the Password Management server (**pwd_strength**). The value of the variable is set to **-v** (verbose) at server configuration.

Variable	Value
REMOVE_PREV_INSTALL	An indication of whether or not to remove all remnants of previous DCE installations before performing the new install: **y** indicates remove all remnants; **n** indicates do not. Be aware that if you set this variable to O, **dce_config** will automatically remove all installed components each time you install any component, and you must reinstall them all. Used in all component installations.
REMOVE_PREV_CONFIG	An indication of whether or not to remove all remnants of previous configurations before performing the new configuration: **y** indicates remove all remnants; **n** indicates do not. Be aware that if you set this variable to O, **dce_config** will stop and remove all configured components each time you configure any component, and you must reconfigure them all. Used in all component configurations.
REP_CLEARINGHOUSE	The name for new clearinghouse. Used in additional CDS server configuration.
SEC_SERVER	The name of the machine on the the cell's master Security server runs. Used in security client configuration.
SEC_SERVER_IP	The IP address for server named in SEC_SERVER.

Variable	Value
SYNC_CLOCKS	An indication of whether or not to synchronize all client clocks with the Security server clock: **y** indicates that client and server clocks will be synchronized; **n** indicates they will not. If this variable is set to **n**, and clocks are out of sync by more than the value specified in the TOLERANCE_SEC variable, the user is prompted for whether or not to synchronize them. This variable is valid only if the CHECK_TIME variable is set to **y**. For DFS machine configurations, this variable should be set to **y**.
TIME_SERVER	Specifies the host that the Security client will try to synchronize its clock against. This host must have a DTS server (**dtsd**) running on it. The recommended choice for the host is the one running the master Security server (the name specified in the SEC_SERVER variable).
TOLERANCE_SEC	The number of seconds a client system clock can differ from the Security server system clock before either the user prompted to synchronize clocks or clocks are synchronized automatically. The default is 120 seconds. Both the Security service and the CDS service require that be no more than a 5-minute difference between the clocks on any two nodes in a cell. For a DFS File Location Database Server, the variable should not be set to less than 90 seconds.

Variable	Value
UID_GAP	The increment above highest currently used UID at which the Security service will start assigning automatically generated UIDs. The value of this variable is used with the LOW_UID variable to set the starting point for UIDs automatically assigned by the Security server. Default is 100. Used in Security server configuration.
UNCONFIG_HOST_PRESET	The name of the node to be unconfigured. Used with the **unconfigure** option.

10.2.2 The dfs_config Environment Variables

Many variables specific to DFS are set explicitly in the **dfs_config** component script. You can use these variables to supply information to **dfs_config** instead of typing the information in in response to prompts.

Table 10-2 shows the DFS values that are set in the **dfs_config** component script. In the table, the term default refers to the setting assigned to the environment variable by OSF.

Table 10–2. dfs_config Environment Variables

Variable	Value
AGG_FS_TYPE	The type of filesystem for the aggregate to be exported. Possible values are **native** meaning the native file system (e.g. UFS, JFS) or **episode** meaning the Episode (LFS) file system.
AGG_DEV_NAME	The device name of the aggregate to be exported.

Variable	Value
AGG_MOUNT_PATH	The mount path for the aggregate (e.g. /usr/users).
AGG_NAME	The name to be used for the aggregate to be exported (e.g. user.jlw).
AGG_ID	The unique numerical aggregate ID for the exported aggregate.
CACHE_SIZE_RAM	The number of bytes to use for an in-memory cache.
CACHE_SIZE_DISK	The number of bytes to use for a local disk cache.
CACHE_DIR_DISK	The pathname of the directory to use for a local disk cache.
CLIENT_CACHE_LOC	An indication of whether the cache is stored in memory or on disk. Machine values are **mem** meaning the cache is stored in memory or **disk** meaning the cache is stored on the local disk.
CONFIG_NFS_GATEWAY	An indication of whether or not to configure the DFS client as an NFS gateway. Possible values are **y** and **n**; **n** is the default.
DFS_SERVER_INSTALL	An indication of which type of DFS server to install: **SCM** for System Control Machine; **FS** for File Server; **PRIVATEFS** for Private File Server; **FLDB** for File Location Database Server.
EPI_FORMAT_PART	An indication of whether or not to format a disk partition as an Episode aggregate. Possible values are **y** to format the partition or **n** to not.

Variable	Value
EPI_FORCE_INIT	An indication of whether or not to force the initialization of a partition as an Episode aggregate, possibly losing data. Possible values are **y** or the initialization or **n** to not.
INIT_LFS	An indication of whether or not to initialize the LFS (using **epiinit**). Possible values are **y** to initialize or **n** to not.
INSTALL_OPT_SERS	An indication of whether or not to install the optional DFS servers (**bak**, **butc**, **upclient**). Possible values are **y** to install or **n** to not.
INSTALL_OPT_CLIENT	An indication of whether or not to install the optional DFS client (**cm**, **bos**, and **fts**) binaries. Possible values are **y** to install or **n** to not.
LOAD_LFS_KEXT	An indication of whether or not to load the LFS kernel extensions. Possible values are **y** to load or **n** to not.
ROOT_FILESET_NM	The name of the DFS root fileset.
SCM_NAME	The name of the system control machine to be used during configuration.

10.3 Controlling Message Logging

Messages are logged to the display screen (**stdout**) and to the **dce_config** log file, **/tmp/dce_config.log**. Some of the types of messages are always displayed and logged. However, using the environment variables described in Section 10.2, you can control the display and logging of others. Table 10-3 lists the message types and how to control their display or logging.

Table 10–3. Environment Variables and Message Logging

Message		
Type	**Displayed?**	**Logged to file?**
Error	Always	Always
Warning	Always	Always
Summary	Always	If DC_LOG_THRESHOLD environment variable set to **summary**, **detail**, **verbose**, or **debug**.
Detail	If DC_DISPLAY_THRESHOLD environment variable set to **detail**, **verbose**, or **debug**.	If DC_LOG_THRESHOLD environment variable set to **detail**, **verbose**, or **debug**.
Verbose	If LDC_DISPLAY_THRESHOLD environment variable set to **verbose**, or **debug**.	If DC_LOG_THRESHOLD environment variable set to **verbose**, or **debug**.
Debug	If DC_DISPLAY_THRESHOLD environment variable set to **debug**.	If DC_LOG_THRESHOLD environment variable set to **debug**.

10.4 Using the dce_config Component Scripts

The **dce_config** script calls component scripts that reside in the **/opt/dcelocal/etc** directory (or the install tree directory) with symbolic links to **/etc**. In a custom configuration script, you can call the component scripts directly and supply the required input via the environment variables. The names and functions of the component scripts follows:

- **dce_shutdown**—Kills all DCE server processes except DFS processes. This script must be run on the machine running the processes. It should be run before reconfiguring DCE.

  The **dce_shutdown** script kills the CDS, Security, DTS, Audit, and GDS daemons gracefully. It also has function (the **-f** option of the command syntax) that can

be used to kill these daemons and other daemons ungracefully if the need arises. See the **dce_shutdown** reference page for detailed information.

- **dfs.clean**—Kills DFS server processes. This script must be run on the machine running the processes. It should be run before reconfiguring DCE. (Note that some DFS daemon processes cannot be killed by **dfs.clean**.)
- **dce.rm** [**install**]—Removes all data and configuration files created by DCE servers after initial configuration except for data and files created by DFS servers. This script must be run on the machine running the processes. It should be run before reconfiguring DCE. If you invoke the script with the **install** parameter, the script removes the binary files added during installation.
- **dfs.rm** [**install**]—Removes data and configuration files created by DFS servers after initial configuration. This script must be run on the machine running the processes, and **dced** must be running on that machine. The **dfs.rm** script should be run before reconfiguring DCE. If you invoke the script with the **install** parameter, the script removes the binary files added during installation. Note that this script invokes the **dce.clean** script.
- **dce.unconfig** *hostname*—Removes all DCE clients on *hostname* from the Security and Directory service databases. It should be run before reconfiguring a client machine.
- **dfs.unconfig** *hostname*—Removes the DFS client on *hostname* from the Security and Directory service databases. It should be run before reconfiguring a client machine.
- **dce_com_env**—Sets environment variables.
- **dce_config_env**—Calls the **dce_com_env** script that sets the internal environment variables.
- **dce_com_utils**—Contains common functions used by **dce_config** and **dfs_config**.
- **dce_config_utils**—Contains internal routines used by **dce_config** scripts.
- **dfs_config**—Configures a machine as a DFS server or client.
- **pwd_config [-unconfig]**—Configures the Password Management server on a machine.
- **rc.dce**—Starts DCE daemons. This script cannot be run remotely; it must be run on the machine on which the daemons are being started.

- **rc.dfs**—Starts DCE daemons. This script cannot be run remotely; it must be run on the machine on which the daemons are being started.

Appendix A

The DCE Cell Namespace

This appendix describes the names that CDS and the DCE Security Service use within the DCE cell namespace. These namespace entries are created during initial DCE configuration.

In the tables that follow, the CDS Class field is either used internally by the **CDS_Clearinghouse** entry and the RPC NSI. The Well Known field specifies whether the last component of a name is an architecturally required name. The Default ACLS field specifies the ACLs created by running the DCE configuration script.

The *hostname*, *lclhostname*, *cellname*, and *creator* entries are defined as follows:

- *hostname*

  This is a cell-relative hostname. For example, the *hostname* for a host named **machine1.abc.com** is **machine1**. Note that for cells with subdomains, a directory structure is possible. For example, the host **apollo.mercury.acs.cmu.edu** can have a *hostname* of **acs/mercury/apollo**.

- *lclhostname*

This is the single component hostname. This name is always the least significant component of the hostname. The *lclhostname* for the examples given previously are **machine1** and **apollo**.

- *cellname*

  This is the global name of the cell, without the special character string **/.../**; for example, **seattle.abc.com** or **C=US/O=ABC/OU=Seattle**.

- *creator*

  This is the name of the principal that created the cell.

A.1 The CDS Space

Figures A-1 through A-3 illustrate the CDS namespace of a DCE cell namespace. The subsections that follow provide a description of each entry.

Figure A–1. The Top-Level CDS Directory

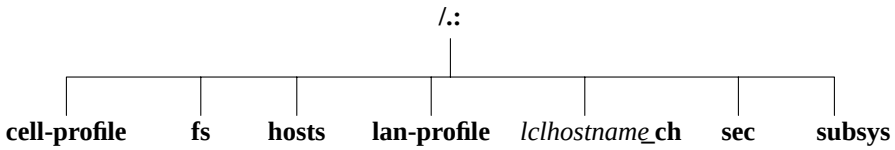


Figure A–2. The CDS hosts Directory

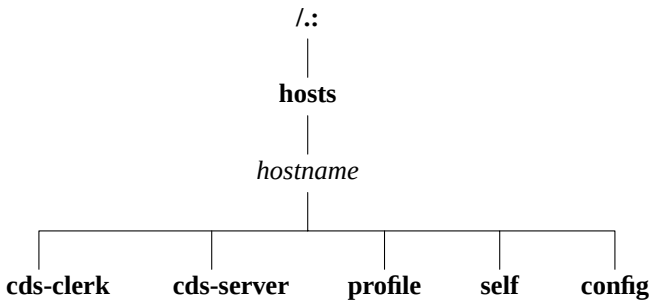
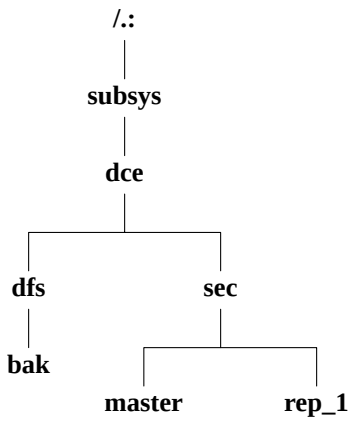


Figure A–3. The CDS subsys Directory



A.1.1 The Top-Level CDS Directory

The following tables describe the namespace entries for **/.:**, the top-level CDS directory.

Name	**/.:**
CDS Type	Directory
Well Known	Yes
Description	This is the cell root directory. The special character string **/.:** is a shorthand form of **/.../** *cellname*. This directory is replicated in every clearinghouse.
Default ACLs	
Object ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**
Initial Object ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtc––}** **{group subsys/dce/cds-server rwdtc––}** **{any_other r––t–––}**
Initial Container ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**

Name	**/.:/cell-profile**
CDS Type	Object
CDS Class	**RPC_Profile**
Well Known	Yes
Description	This is the master default profile for the cell. Ultimately, all other profiles should link to this profile. This profile is created at cell creation and must include the following entry: *LAN-Services-UUID /.../cellname/***lan-profile** Note that like all profile entries, only global names can be used. This profile must include interfaces for the Privilege Server, the Registry Server, and the Authentication Server. In multi-LAN cells this is the profile in which the DTS global set entries are entered.
Default ACLs	
Object ACL	**{unauthenticated r– – t– }** **{user** *creator* **rwdtc}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{group subsys/dce/dts-admin rw– t– }** **{group subsys/dce/dts-servers rw– t– }** **{any_other r– – t– }**

Name	**/.:/fs**
CDS Type	Object
CDS Class	**RPC_Group**
Well Known	No
Description	This is the junction to the DFS filespace within the cell namespace. The character string **/:** is a CDS soft link to **/.:/fs**. The RPC bindings of all Fileset Database machines housing the FLDB are listed in this group. This group consists of RPC entries of the following form: **/.../***cellname***/hosts/***hostname***/self** This object has a single object UUID attached to it.
Default ACLs	
Object ACL	**{unauthenticated r– – t– }** **{user** *creator* **rwdtc}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{group subsys/dce/dfs-fs-servers rwdtc}** **{group subsys/dce/dfs-admin rwdtc}** **{any_other r– – t– }**

Name	**/.:/hosts**
CDS Type	Directory
Well Known	No
Description	The host directories are cataloged here.
Default ACLs	
Object ACL	**{unauthenticated r––t–––}** **{user** *creator* **rwdtcia}** **{user hosts/**_hostname_**/cds-server rwdtcia}** **{user hosts/**_hostname_**/self rwdtcia}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**
Initial Object ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtc––}** **{group subsys/dce/cds-server rwdtc––}** **{any_other r––t–––}**
Initial Container ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**

Name	**/.:/lan-profile**
CDS Type	Object
CDS Class	**RPC_Profile**
Well Known	No
Description	This is the default LAN profile used by DTS, and potentially by other services. In single LAN cells, this is the profile in which entries for the DTS local set entries are entered.
Default ACLs	
Object ACL	**{unauthenticated r– – t– }** **{user** *creator* **rwdtc}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{group subsys/dce/dts-admin rwdtc}** **{group subsys/dce/dts-servers rwdtc}** **{any_other r– – t– }**

Name	**/.:/***lclhostname***_ch**
CDS Type	Object
CDS Class	**CDS_Clearinghouse**
Well Known	No
Description	All clearinghouses are cataloged in the cell root. This name is only fixed for the first CDS Server you configure. You can choose different names for any additional CDS Servers you configure.
Default ACLs	
Object ACL	**{unauthenticated r––t–}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{any_other r––t–}**

Name	**/.:/sec**
CDS Type	Object
CDS Class	**RPC_Group**
Well Known	No
Description	This is the RPC group of all Security Servers for this cell. It contains the entries **/...**/*cellname*/**subsys/dce/sec/master** and (for example) **/.../***cellname*/**subsys/dce/sec/rep_1**. This is the junction into the Security namespace.
Default ACLs	
Object ACL	**{unauthenticated r– – t– }** **{user** *creator* **rwdtc}** **{user dce-rgy rwdtc}** **{user hosts/***rep_1_hostname*/**self rwdtc}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{group subsys/dce/sec-admin rwdtc}** **{any_other r– – t– }**

Name	**/.:/subsys**
CDS Type	Directory
Well Known	No
Description	This directory contains directories for different subsystems in this cell. It contains the **dce** subdirectory. It is recommended that companies adding subsystems to DCE conform to the convention of creating a unique directory below **subsys** by using their trademark as a directory name (**/.:/subsys/***trademark*). These directories are used for storage of location-independent information about services. Server entries, groups, and profiles for the entire cell should be stored in directories below **subsys**.
Default ACLs	
Object ACL	**{unauthenticated r––t–––}** **{user** *creator* **rwdtcia}** **{user hosts/***hostname* **rwdtcia}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**
Initial Object ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtc––}** **{group subsys/dce/cds-server rwdtc––}** **{any_other r––t–––}**
Initial Container ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**

A.1.2 The CDS hosts Directory

The following tables describe the namespace entries for **/.:/hosts**, the CDS **hosts** directory.

Name	**/.:/hosts/***hostname*
CDS Type	Directory
Well Known	No
Description	Each host has a directory in which RPC server entries, groups, and profiles associated with this host are stored. This is simply a CDS directory. No bindings are present in the directory object itself; entries exist beneath the directory.
Default ACLs	
Object ACL	**{unauthenticated r––t–––}** **{user** *creator* **rwdtcia}** **{user hosts/***hostname***/cds-server rwdtcia}** **{user hosts/***hostname***/self rwdtcia}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**
Initial Object ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtc––}** **{group subsys/dce/cds-server rwdtc––}** **{any_other r––t–––}**
Initial Container ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**

Name	**/.:/hosts/***hostname***/cds-clerk**
CDS Type	Object
CDS Class	**RPC_Entry**
Well Known	No
Description	This entry contains the binding for a CDS clerk.
Default ACLs	
Object ACL	**{unauthenticated r––t–}** **{user** *creator* **rwdtc}** **{user hosts/***hostname***/self rw–t–}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{any_other r––t–}**

Name	**/.:/hosts/***hostname***/cds-server**
CDS Type	Object
CDS Class	**RPC_Entry**
Well Known	No
Description	This entry contains the binding for a CDS Server.
Default ACLs	
Object ACL	**{unauthenticated r––t–}** **{user** *creator* **rwdtc}** **{user hosts/***hostname***/self rw–t–}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{any_other r––t–}**

Name	**/.:/hosts/***hostname***/config**
CDS Type	Object
CDS Class	**RPC_Entry**
Well Known	Yes
Description	This is the server entry for the **dced** on the given host. It is also the top of the naming tree for that **dced**. Programs obtain this name by using the call **dce_cf_dced_entry_from_host( )**.
Default ACLs	
Object ACL	**{unauthenticated r––t–––}** **{user hosts/***hostname***/self rwdtc––}** **{group subsys/dce/cds-admin rwdtc––}** **{group subsys/dce/cds-server rwdtc––}** **{any_other r––t–––}**

Name	**/.:/hosts/***hostname***/profile**
CDS Type	Object
CDS Class	**RPC_Entry**
Well Known	No
Description	This is the default profile for host *hostname*. It must contain a default that points (possibly indirectly) at **/.:/cell-profile**. Programs obtain this name by using the call **dce_cf_profile_entry_from_host( )**.
Default ACLs	
Object ACL	**{unauthenticated r– – t– }** **{user** *creator* **rwdtc}** **{user hosts/***hostname***/self rw– t– }** **{group subsys/dce/cds-admin rwcdt}** **{group subsys/dce/cds-server rwcdt}** **{any_other r– – t– }**

Name	**/.:/hosts/***hostname***/self**
CDS Type	Object
CDS Class	**RPC_Entry**
Well Known	Yes
Description	This entry contains a binding to the **dced** daemon on host *hostname*. The **dce_cf_binding_entry_from_host( )** call returns either the name of this entry when handed a hostname or the current host when a hostname is not provided.
Default ACLs	
Object ACL	**{unauthenticated r– – t– }** **{user** *creator* **rwdtc}** **{user hosts/***hostname***/self rwrdtc}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{any_other r– – t– }**

A.1.2.1 The Host Daemon Directory

The following tables describe the **dced** namespace entries for **/.:/hosts/***hostname***/config**, the host daemon directory. These are all created by **dced** as part of configuration.

Name	**/.:/hosts/***hostname***/config**
dced Type	**dced** object
Well Known	Yes
Description	The **dced** server itself.
Default ACLs	
Object ACL	**{user hosts/***hostname***/self crws}**

Name	**/.:/hosts/***hostname***/config/hostdata**
dced Type	**dced** container
Well Known	Yes
Description	The container for **hostdata** objects on the given host.
Default ACLs	
Object ACL	**{user hosts/***hostname***/self criI}**
Initial Object ACL	**{user hosts/***hostname***/self cdprw}**

Name	**/.:/hosts/***hostname***/config/keytab**
dced Type	**dced** container
Well Known	Yes
Description	The container for **keytab** objects on the given host.
Default ACLs	
Object ACL	**{user hosts/***hostname***/self criI}**
Initial Object ACL	**{user hosts/***hostname***/self acdepr}**

Name	**/.:/hosts/***hostname***/config/secval**
dced Type	**dced** object
Well Known	Yes
Description	The name of the **secval** service.
Default ACLs	
Object ACL	**{user hosts/***hostname***/self csux}**

Name	**/.:/hosts/***hostname***/config/srvrconf**
dced Type	**dced** container
Well Known	Yes
Description	Container for the configured servers registered with **dced**.
Default ACLs	
Object ACL	**{user hosts/***hostname***/self criI}**
Initial Object ACL	**{user hosts/***hostname***/self cdfrwx}**

Name	**/.:/hosts/***hostname***/config/srvrexec**
dced Type	**dced** container
Well Known	Yes
Description	Container for the running servers registered with **dced**.
Default ACLs	
Object ACL	**{user hosts/***hostname***/self criI}**
Initial Object ACL	**{user hosts/***hostname***/self crws}**

Name	**/.:/hosts/***hostname***/config/xattrschema**
dced Type	**dced** container
Well Known	Yes
Description	The container of extended attribute schema definitions.
Default ACLs	
Object ACL	**{user hosts/***hostname***/self criI}**
Initial Object ACL	**{user hosts/***hostname***/self crwd}**

A.1.3 The CDS subsys Directory

The following tables describe the namespace entries for **/.:/subsys**, the CDS **subsys** directory.

Name	**/.:/subsys/dce**
CDS Type	Directory
Well Known	No
Description	This directory contains DCE-specific names.
Default ACLs	
Object ACL	**{unauthenticated r––t–––}** **{user** *creator* **rwdtcia}** **{user hosts/**_hostname_**/cds-server rwdtcia}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**
Initial Object ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtc––}** **{group subsys/dce/cds-server rwdtc––}** **{any_other r––t–––}**
Initial Container ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{any_other r––t–––}**

Name	**/.:/subsys/dce/dfs**
CDS Type	Directory
Well Known	No
Description	This directory contains all of the DFS-specific names.
Default ACLs	
Object ACL	**{unauthenticated r––t–––}** **{user** *creator* **rwdtcia}** **{user hosts/***hostname***/cds-server rwdtcia}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{group subsys/dce/dfs-admin rwdtcia}** **{any_other r––t–––}**
Initial Object ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtc––}** **{group subsys/dce/cds-server rwdtc––}** **{group subsys/dce/dfs-admin rwdtc––}** **{any_other r––t–––}**
Initial Container ACL	**{unauthenticated r––t–––}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{group subsys/dce/dfs-admin rwdtcia}** **{any_other r––t–––}**

Name	**/.:/subsys/dce/dfs/bak**
CDS Type	Object
CDS Class	**RPC_Entry**
Well Known	No
Description	The RPC bindings of all Backup Database machines that are storing the Backup Database are listed in this entry. This entry is similar to the **/.:/fs** group in that its members are RPC entries of the **/.../***cellname***/hosts/***hostname***/self** form. In addition, this group must have a single object UUID attached to it.
Default ACLs	
Object ACL	**{unauthenticated r– – t– }** **{user** *creator* **rwdtc}** **{user hosts/***hostname***/cds-server rwdtc}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{any_other r– – t– }**

Name	**/.:/subsys/dce/sec**
CDS Type	Directory
Well Known	No
Description	This directory contains Security-specific names.
Default ACLs	
Object ACL	**{unauthenticated r––t–––}** **{user** *creator* **rwdtcia}** **{user hosts/***hostname***/cds-server rwdtcia}** **{user dce-rgy rwdtci–}** **{user hosts/***rep_1_hostname***/self rwdtia}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{group subsys/dce/sec-admin rwdtcia}** **{any_other r––t–––}**
Initial Object ACL	**{unauthenticated r––t–––}** **{user dce-rgy rwdt–––}** **{user hosts/***rep_1_hostname***/self rwdtc}** **{group subsys/dce/cds-admin rwdtc––}** **{group subsys/dce/cds-server rwdtc––}** **{group subsys/dce/sec-admin rwdtc––}** **{any_other r––t–––}**
Initial Container ACL	**{unauthenticated r––t–––}** **{user dce-rgy rwdtci–}** **{user hosts/***rep_1_hostname***/self rwdtcia}** **{group subsys/dce/cds-admin rwdtcia}** **{group subsys/dce/cds-server rwdtcia}** **{group subsys/dce/sec-admin rwdtcia}** **{any_other r––t–––}**

Name	**/.:/subsys/dce/sec/master**
CDS Type	Object
CDS Class	**RPC_Entry**
Well Known	No
Description	This is the server entry for the master Security Server for this cell. The bindings for the Registry Server, the Privilege Server, and the Authentication Server are exported by the Registry Server to this entry.
Default ACLs	
Object ACL	**{unauthenticated r– – t– }** **{user dce-rgy rwdt– }** **{user** *creator* **rwdtc}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{group subsys/dce/sec-admin rwdtc}** **{any_other r– – t– }**

Name	**/.:/subsys/dce/sec/rep_1**
CDS Type	Object
CDS Class	**RPC_Entry**
Well Known	No
Description	This is the server entry for a slave Security Server for this cell. The bindings for the Registry Server, the Privilege Server, and the Authentication Server are exported by the Registry Server to this entry.
Default ACLs	
Object ACL	**{unauthenticated r–t–}** **{user dce-rgy rwdt-}** **{user** *creator* **rwdtc }** **{user hosts/***rep_1_hostname***/self rwdtc}** **{group subsys/dce/cds-admin rwdtc}** **{group subsys/dce/cds-server rwdtc}** **{group subsys/dce/sec-admin rwdtc}** **{any_other r–t–}**

A.2 The Security Space

Figures A-4 through A-6 illustrate the Security namespace within the DCE cell namespace. The subsections that follow provide a description of each entry. The subdirectories that comprise the Security namespace are **principal**, **group**, **org**, **policy**, **replist**, and **xattrschema**.

To operate on the ACLs on any of these namespace entries, you need to include the name of the Security junction. For example, when you use the DCE control program's (**dcecp**) **acl** commands, the group name **acct-admin** is referenced as **/.:/sec/group/acct-admin**, its database object name.

However, when you use the **dcecp principal**, **group**, or **organization** commands, operate on a principal, group, or organization name without **/.:/sec** and **principal**,

group, or **organization** included as part of the name. For example, to view the attributes of the group **acct-admin**, you issue the **group show** command specifying the group name **acct-admin** without this path.

Figure A–4. The Top-Level Security Directory

Figure A–5. The sec/group Directory

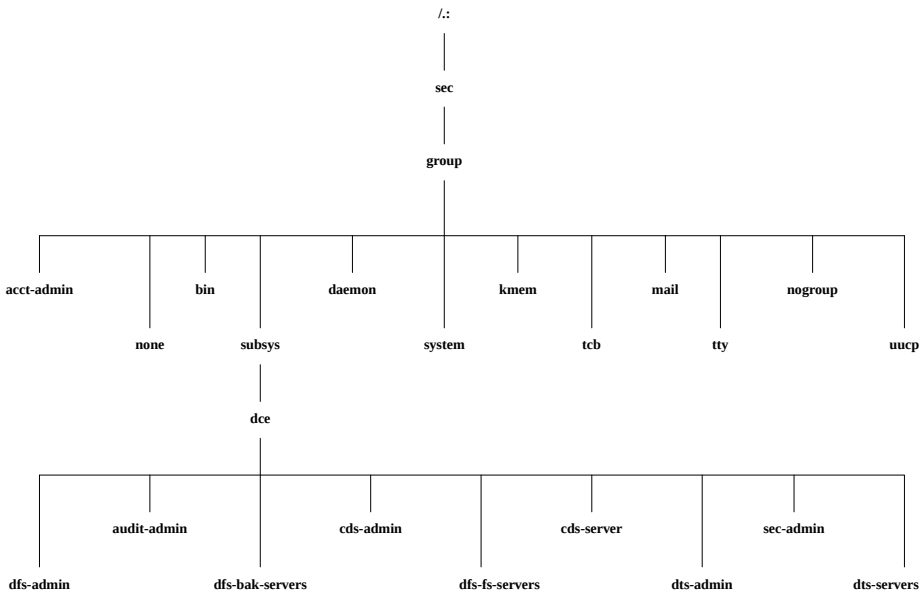
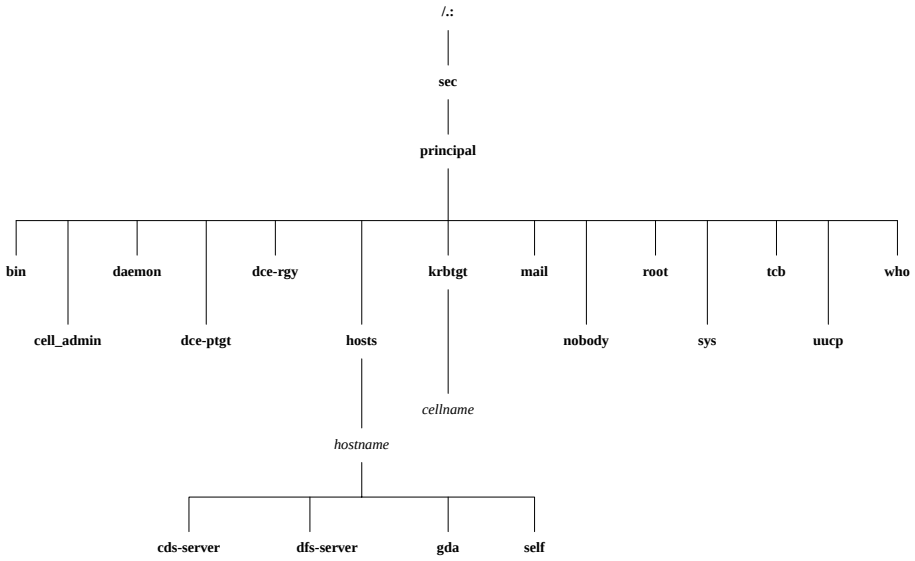


Figure A–6. The sec/principal Directory



In the following subsections, descriptions of entries in an initial Security namespace are given. Included is the suggested UNIX user identifier (UNIX UID) or group identifier (UNIX GID) that they are assigned to. Vendors should use these values if possible. The password and group override files can replace them with correct local values, if necessary. Some entries are assigned the next available identifier, starting with 100; therefore, these may vary from cell to cell. They are indicated as ‘‘Generated’’.

A.2.1 The Top-Level Security Directory

The following tables describe the namespace entries for **/.:/sec**, the top-level Security directory.

Name	**/.:/sec/group**
Well Known	Yes. This name is not architecturally defined, but is defined by the implementation.
Description	This is the Security directory that holds all the groups.
Default ACLs	
Object ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**
Initial Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–––––––}**
Initial Container ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**

Name	**/.:/sec/org**
Well Known	Yes. This name is not architecturally defined, but is defined by the implementation.
Description	This is the Security directory that holds all the organizations.
Default ACLs	
Object ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**
Initial Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
Initial Container ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**

Name	**/.:/sec/org/none**
Well Known	Yes
Description	This is the default organization.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**

Name	**/.:/sec/policy**
Well Known	Yes. This name is not architecturally defined, but is defined by the implementation.
Description	This entry provides the ability to set Security policies on a cell-wide basis.
Default ACLs	
Object ACL	**{unauthenticated r––––}** **{user** *creator* **rcmaA}** **{group acct-admin rcmaA}** **{other_obj r––––}** **{any_other r––––}**

Name	**/.:/sec/principal**
Well Known	Yes. This name is not architecturally defined, but it cannot be changed in DCE 1.1.
Description	This is the Security directory that holds all of the principals.
Default ACLs	
Object ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other_obj r–––––}**
Initial Object ACL	**{unauthenticated r–––––––g}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
Initial Container ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**

Name	**/.:/sec/replist**
Well Known	Yes. This name is not architecturally defined, but it cannot be changed in DCE 1.1.
Description	This entry holds information about the different security replicas.
Default ACLs	
Object ACL	**{user** *creator* **cidmA–}** **{user hosts/***hostname***/self –i–m–I}** **{group acct-admin cidmA–}**

Name	**/.:/sec/xattrschema**
Well Known	Yes. This name is not architecturally defined, but it cannot be changed in DCE 1.1.
Description	This is a container for extended registry attribute schema entries. The entries within this directory define the format of ERAs that may be attached to other registry objects (for example, principals).
Default ACLs	
Object ACL	**{unauthenticated r––––}** **{user** *creator* **rcidm}** **{other_obj r––––}** **{any_other r––––}**

A.2.2 The sec/group Directory

The following tables describe the namespace entries for **/.:/sec/group**, the Security **sec/group** directory.

Name	**/.:/sec/group/acct-admin**
Well Known	No

Description	This is the only group of principals that can create accounts.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/bin**
Well Known	No
Description	This is the group for system binaries.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	3

Name	**/.:/sec/group/daemon**
Well Known	No
Description	This is the group for daemons.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	1

Name	**/.:/sec/group/kmem**
Well Known	No
Description	This is the group that has read access to kernel memory.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	4

Name	**/.:/sec/group/mail**
Well Known	No
Description	This is the group for the mail subsystem.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	6

Name	**/.:/sec/group/nogroup**
Well Known	Yes
Description	This is the default group for NFS access; it goes with user ID **nobody**.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	-2

Name	**/.:/sec/group/none**
Well Known	Yes
Description	This member does not belong to a group; it is the default group.
Default ACLs	
Object ACL	**{unauthenticated r– t– – – – –}** **{user** *creator* **rctDnfmM}** **{group_obj r– t– – – – –}** **{group acct-admin rctDnfmM}** **{other_obj r– t– – – – –}** **{any_other r– t– – – – –}**
UNIX GID	12

Name	**/.:/sec/group/subsys**
Well Known	Yes
Description	This directory contains **dce**. (See **/.:/subsys** in the CDS namespace.)
Default ACLs	
Object ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**
Initial Object ACL	**{unauthenticated r–t–––––}** **{user creator rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
Initial Container ACL	**{unauthenticated r–––––}** **{user creator rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**

Name	**/.:/sec/group/system**
Well Known	No
Description	This is the group for system accounts.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	0

Name	**/.:/sec/group/tcb**
Well Known	No
Description	This is the group used by security policy daemons on OSF/1 C2/B1 secure systems.
Default ACLs	
Object ACL	**{unauthenticated r-t—–}** **{user** *creator* **rctDnfmM}** **{group_obj r-t—–}** **{group acct-admin rctDnfmM}** **{other_obj r-t—–}** **{any_other r-t—–}**
UNIX GID	18

Name	**/.:/sec/group/tty**
Well Known	No
Description	This is the group that has write access to terminals.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	7

Name	**/.:/sec/group/uucp**
Well Known	No
Description	This is the group for the UUCP subsystem.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	2

A.2.3 The sec/group/subsys Directory

The following tables describe the namespace entries for **/.:/sec/group/subsys**, the Security **sec/group/subsys** directory.

Name	**/.:/sec/group/subsys/dce**
Well Known	Yes
Description	This directory contains the groups used by DCE.
Default ACLs	
Object ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**
Initial Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–rt–––––}** **{group acct-admin rcitDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
Initial Container ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**

Name	**/.:/sec/group/subsys/dce/cds-admin**
Well Known	No
Description	This is the administrative group that is on the default ACLs for administrative objects. Clearinghouses have this group on their ACLs with all rights. The first user of the cell must be added to this group immediately after creation.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/subsys/dce/cds-server**
Well Known	Yes
Description	This is the group of all CDS Servers for the local cell. As each new server is added to the cell, it must be added to this group. CDS Server authentication consists of checking for the server's membership in this group.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{group subsys/dce/cds-admin rctDnfmM}** **{group subsys/dce/cds-server rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/subsys/dce/dfs-admin**
Well Known	No
Description	This is the DFS administrator's group. Members of this group have full permissions to alter the DFS configuration within the cell.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/subsys/dce/dfs-bak-servers**
Well Known	Yes
Description	This is the Security group to which all DFS Backup Database Servers belong. A server entry in the CDS group **/.:/subsys/dce/fs** is checked for authorization to act as a Backup Database Server by determining whether it belongs to this Security group.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/subsys/dce/dfs-fs-servers**
Well Known	Yes
Description	Abbreviated forms of the DFS Server principals of all Fileset Database machines are listed in this group. The abbreviated form of a machine's DFS Server principal stored in the group is of the form **hosts/***hostname***/dfs-server**. A server entry obtained from the CDS group **/.:/fs** is checked for authorization to act as a Fileset Location Server by determining if it belongs to this group.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{group subsys/dce/dfs-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/subsys/dce/dts-admin**
Well Known	No
Description	This is the DTS administrator's group. Members of this group have full permissions to administer DTS by adding servers and so forth.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/subsys/dce/dts-servers**
Well Known	Yes
Description	This is the group of DTS Servers.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{group subsys/dce/dts-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/subsys/dce/sec-admin**
Well Known	No
Description	This is the Security administrator's group. Members of this group have full permissions to administer the Security database.
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

Name	**/.:/sec/group/subsys/dce/audit-admin**
Well Known	No
Description	This is the Audit daemon administrator's group. Members of this group have full permissions to administer the Audit daemon (**auditd**).
Default ACLs	
Object ACL	**{unauthenticated r–t–––––}** **{user** *creator* **rctDnfmM}** **{group_obj r–t–––––}** **{group acct-admin rctDnfmM}** **{other_obj r–t–––––}** **{any_other r–t–––––}**
UNIX GID	Generated

A.2.4 The sec/principal Directory

The following tables describe the namespace entries for **/.:/sec/principal**, the Security **sec/principal** directory.

Name	**/.:/sec/principal/bin**
Well Known	No
Description	This is the owner of the system binaries.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	3

Name	**/.:/sec/principal/cell_admin**
Well Known	No
Description	This is the principal who does the initial cell configuration.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj rcDnfmaug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	Generated

Name	**/.:/sec/principal/daemon**
Well Known	No
Description	This is the user for the various daemons.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	1

Name	**/.:/sec/principal/dce-ptgt**
Well Known	Yes
Description	This is the architecturally defined principal name of the Privilege Server.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	20

Name	**/.:/sec/principal/dce-rgy**
Well Known	Yes
Description	This is the architecturally defined principal name of the Registry Server.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	21

Name	**/.:/sec/principal/hosts**
Well Known	No
Description	This directory contains all DCE host principals.
Default ACLs	
Object ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**
Initial Object ACL	**{unauthenticated r–––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r–––––––––}**
Initial Container ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**

Name	**/.:/sec/principal/krbtgt** *(also known as* **/...***)*
Well Known	Yes
Description	This is the architecturally specified name of the Security namespace where foreign cell names are cataloged. All cells that this cell communicates with appear here.
Default ACLs	
Object ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**
Initial Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
Initial Container ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**

Name	**/.:/sec/principal/krbtgt/***cellname (also known as /.:)*
Well Known	No
Description	This is the principal of the Authentication Server of the cell named **/.../***cellname*. In the local cell, this is the principal for **/.:**.
Default ACLs	
Object ACL	**{unauthenticated r–––––––g}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**

Name	**/.:/sec/principal/mail**
Well Known	No
Description	This is the user for the mail subsystem.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	6

Name	**/.:/sec/principal/nobody**
Well Known	No
Description	This is the default user for NFS access.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	2

Name	**/.:/sec/principal/root**
Well Known	No
Description	This is the local operating system superuser.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	0

Name	**/.:/sec/principal/sys**
Well Known	No
Description	This is a user who is permitted to read devices but is not a superuser.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	2

Name	**/.:/sec/principal/tcb**
Well Known	No
Description	This is the user for security policy daemons on OSF/1 C2/B1 secure systems.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	9

Name	**/.:/sec/principal/uucp**
Well Known	No
Description	This is the user for the UUCP subsystem.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	4

Name	**/.:/sec/principal/who**
Well Known	No
Description	This is the user for remote **who** access.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	5

A.2.5 The sec/principal/hosts Directory

The following tables describe the namespace entries for **/.:/sec/principal/hosts**, the Security **sec/principal/hosts** directory.

Name	**/.:/sec/principal/hosts/***hostname*
Well Known	No
Description	This directory contains Security principals for host *hostname*.
Default ACLs	
Object ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**
Initial Object ACL	**{unauthenticated r–––––––g}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
Initial Container ACL	**{unauthenticated r–––––}** **{user** *creator* **rcidDn}** **{group acct-admin rcidDn}** **{other_obj r–––––}** **{any_other r–––––}**

Name	**/.:/sec/principal/hosts/***hostname***/cds-server**
Well Known	No
Description	The CDS Server on node *hostname* runs as this principal. This principal must be a member of the **/.:/sec/group/subsys/dce/cds-server** security group.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfma–g}** **{group subsys/dce/cds-admin rcDnfma–g}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	Generated

Name	**/.:/sec/principal/hosts/***hostname***/dfs-server**
Well Known	No
Description	This is the principal name of the DFS Servers on node *hostname*.
Default ACLs	
Object ACL	**{unauthenticated r–––––––g}** **{user_obj r–––f––ug}** **{user creator rcDnfmaug}** **{group acct_admin rcDnfma–g}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	Generated

Name	**/.:/sec/principal/hosts/***hostname***/gda**
Well Known	No
Description	The GDA on node *hostname* runs as this principal. This principal must be a member of the **/.:/sec/group/subsys/dce/cds-servers** security group.
Default ACLs	
Object ACL	**{unauthenticated r–––––––g}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfmaug}** **{group acct-admin rcDnfmaug}** **{group subsys/dce/cds-admin rcDnfmaug}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	Generated

Name	**/.:/sec/principal/hosts/***hostname***/self**
Well Known	Yes
Description	This entry is the principal for host *hostname*. The security validation service of the **dced** daemon uses this principal. This is also the identity that local root processes can inherit.
Default ACLs	
Object ACL	**{unauthenticated r––––––––}** **{user_obj r–––f––ug}** **{user** *creator* **rcDnfma–g}** **{group acct-admin rcDnfma–g}** **{other_obj r–––––––g}** **{any_other r––––––––}**
UNIX UID	Generated

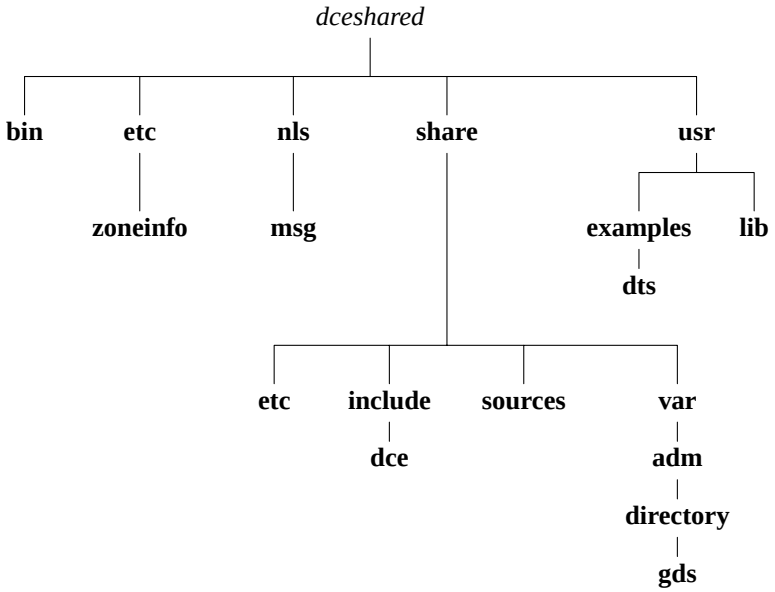
Appendix B
The Location of Installed DCE Files

This appendix shows the organization of the *dceshared*, *dcelocal*, and the UNIX subdirectories used by DCE.

B.1 The dceshared Subdirectories

Figure B-1 shows the *dceshared* subtree.

Figure B–1. The *dceshared* Subtree



The following directories are created in the *dceshared* subtree during installation.

- *dceshared*/**bin**

  This directory contains utilities for applications programmers and DCE users, DCE administration utilities, and server processes (daemons).

- *dceshared*/**etc**

  This directory contains templates of configuration files that are in architecture-dependent format.

- *dceshared*/**etc/zoneinfo**

  This directory contains templates of configuration tables.

- *dceshared*/**nls/msg/${LANG}**

  This directory contains delivered message catalogs (***.cat**) files for each supported language.

- *dceshared*/**share**

  All of the previously described subdirectories can contain architecture-dependent files, which are addressable by using **@sys**. However, the files that are listed after *dceshared*/**share** are completely architecture independent.

- *dceshared*/**share/etc**

  This directory contains templates of common (shared) configuration files.

- *dceshared*/**share/include**

  This directory contains application header files and DCE internal header files. The **/usr/include/dce** directory is linked to this entire directory, but in future DCE releases it could be separated and linked only to those files that are necessary for writing DCE-based applications.

- *dceshared*/**share/sources**

  This directory contains DCE sources and build tools as organized in the Open Development Environment (ODE) build tree, which is available to DCE source licensees only.

- *dceshared*/**usr/examples**

  This directory contains example executable files.

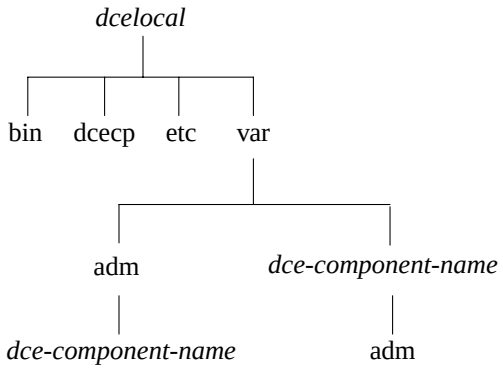
- *dceshared*/**usr/lib**

  This directory contains application libraries (**libdce.a**) and DCE internal libraries.

B.2 The dcelocal Subdirectories

Figure B-2 shows the *dcelocal* subtree.

Figure B–2. The *dcelocal* Subtree



The following directories are created in the *dcelocal* subtree during installation.

- *dcelocal*/**bin**

  This directory contains DCE administration utilities and server processes (daemons), which are necessary for local client system initialization and for server machines.

- *dcelocal*/**dcecp**

  This directory contains **dcecp** scripts.

- *dcelocal*/**etc**

  This directory contains local (machine-private) configuration files, which are maintained by client machines. This directory has write permission for the local system administrator only.

- *dcelocal*/**var**/**adm**/*dce-component-name*

  This directory contains log files (including core images) and cache files maintained by client machines. For convenience, symbolic links from **/var/adm/dce/client/** *dce-component-name* are created. This directory has write permission for the local system administrator only.

- *dcelocal*/**var**/*dce-component-name*

This directory contains all data files (configuration files, databases, and so forth) that are maintained by each of the DCE servers. To provide high availability and (in case of the Security Service) appropriate protection, data files associated with a service are usually physically located at the server site. Therefore, they are stored in separate trees under *dcelocal*/**var**.

Files in *dcelocal*/**var**/*dce-component-name* are only those that are accessed by dedicated servers. This directory has write permission for the service administrator only.

Configuration and log files relative to client machines are not stored here. These files are in *dcelocal*/**etc** and *dcelocal*/**var/adm**.

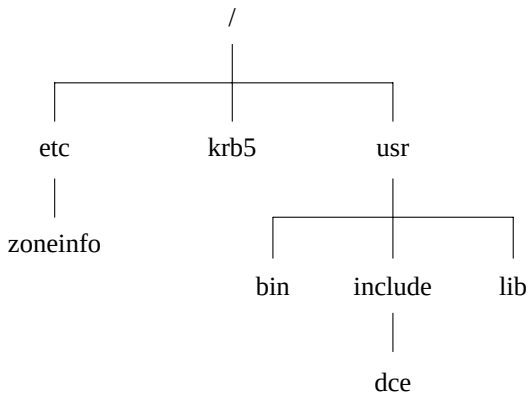
- *dcelocal*/**var**/*dce-component-name*/**adm**

  This directory contains server log files and cache files, which are maintained by server machines. This subdirectory needs to be maintained by each service to store the log and cache files. Because users sometimes expect log files in conventional locations, **/var/adm/dce/***dce-component-name* is created as a symbolic link to these directories. This directory has write permission for the service administrator only.

B.3 Conventional UNIX Directories

Figure B-3 shows the directories that DCE uses in the standard UNIX tree.

Figure B–3. Standard UNIX Directories Tree



DCE uses the following standard UNIX directories.

- **/etc/zoneinfo**

  This directory contains copies of the templates, which are modified, if necessary, from the *dceshared*/**etc**/**zoneinfo** directory.

  Note: Preexisting files can be modified on the local system. Be careful not to overwrite them during the installation procedure.

- **/krb5**

  This directory contains Kerberos configuration files for the conventional **krb5** environment. Symbolic links exist to appropriate files in *dcelocal*/**etc**. This directory has write permission for the local system administrator only.

- **/usr/bin**

  This directory contains utilities for application programmers and DCE users. Most of these are symbolic links that point to *dceshared*/**bin**. Some utilities, such as **login** and **su**, can actually be local copies that are needed for performance and high availability. On server machines, copies of the respective executables are sometimes necessary for the initialization of the system.

- **/usr/include/dce**

This directory contains DCE header files. This directory is a symbolic link to *dceshared*/**share/include/dce**.

- **/usr/lib**

  This directory contains **libdce.a**, which is a symbolic link to *dceshared*/**lib/libdce.a**.

Index

A

B

C

D

E

F

G

H

H

H

I

J

K

L

M

N

P

R

S

S

S

T

U

V

Z