IBM Distributed Computing Environment for AIX,
Version 2.2:

Administration Guide–Introduction

IBM Distributed Computing Environment for AIX,
Version 2.2:

Administration Guide–Introduction

Note

Before using this document, read the general information under “Notices” on page 89.

First Edition (February 1998)

This edition applies to Version 2.2 of the *Distributed Computing Environment for AIX* and to all subsequent releases and modifications until otherwise indicated in new editions or technical newsletters.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. Send your comments to the following address:

International Business Machines Corporation

Department VLXA

11400 Burnet Road

Austin, Texas

78758

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

About This Book

The *IBM DCE for AIX, Version 2.2: Administration Guide* provides concepts and procedures that enable you to manage the IBM AIX version of the Distributed Computing Environment (DCE). Basic DCE terms are introduced throughout the guide. A glossary for all of the DCE documentation is provided in the *IBM DCE for AIX, Version 2.2: Introduction to DCE*. The *IBM DCE for AIX, Version 2.2: Introduction to 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 IBM Distributed Computing Environment for AIX, Version 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.

Document Usage

The *IBM DCE for AIX, Version 2.2: Administration Guide* consists of two books:

- The *IBM DCE for AIX, Version 2.2: Administration Guide—Introduction*
  - “Part 1. Introduction to DCE System Administration” on page 1
  - “Part 2. Additional Configuration Information” on page 39
- The *IBM DCE for AIX, Version 2.2: 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:

- *IBM DCE for AIX, Version 2.2: Quick Beginnings*
- *IBM DCE for AIX, Version 2.2: Introduction to DCE*

- *IBM DCE for AIX, Version 2.2: Administration Commands Reference*
- *IBM DCE for AIX, Version 2.2: Application Development Reference*
- *IBM DCE for AIX, Version 2.2: Application Development Guide—Introduction and Style Guide*
- *IBM DCE for AIX, Version 2.2: Application Development Guide—Core Components*
- *IBM DCE for AIX, Version 2.2: Application Development Guide—Directory Services*
- *IBM DCE for AIX, Version 2.2: DFS Administration Guide and Reference*
- *OSF DCE/File-Access Administration Guide and Reference*
- *OSF DCE/File-Access User's Guide*
- *IBM DCE for AIX, Version 2.2: Problem Determination Guide*
- *OSF DCE Testing Guide*
- *OSF DCE/File-Access FVT User's Guide*
- *Application Environment Specification/Distributed Computing*
- *IBM DCE for AIX, Version 2.2: Release Notes*

For a detailed description of IBM DCE 2.2 for AIX documentation, see the *IBM DCE for AIX, Version 2.2: Introduction to 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.

dcelocal
The OSF variable *dcelocal* in this document equates to the AIX variable **/opt/dcelocal**.

dceshare
The OSF variable *dceshare* in this document equates to the AIX variable **/opt/dcelocal**.

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 "Chapter 4. Location of Installed DCE Files" on page 29, and the *OSF DCE Testing Guide*.

Part 1. Introduction to DCE System Administration

Chapter 1. Introduction to DCE for Administrators

The *IBM DCE for AIX, Version 2.2: Introduction to DCE* introduced you to the IBM AIX version of Distributed Computing Environment (IBM DCE 2.2 for AIX), 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 *IBM DCE for AIX, Version 2.2: Introduction to 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.

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 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.

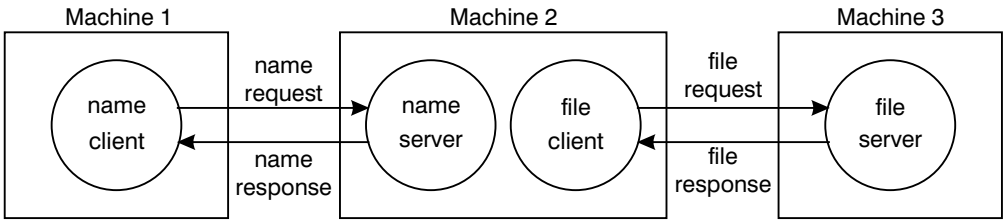

Figure 1. Interaction of Clients and Servers

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.

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.

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. 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.

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.

See the *IBM DCE for AIX, Version 2.2: Quick Beginnings* and *IBM DCE for AIX, Version 2.2: DFS Administration Guide and Reference* for cell filespace planning guidelines.

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. See "Establishing Trust Relationships" in *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components* for information on creating these special accounts.

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)

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.

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.

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, and DFS all provide replication facilities that are customized for their particular applications.

Environment Variables

Environment variables are variables used by DCE that customers can set themselves. See the "Appendix B. Environment Variables" on page 49 for more comprehensive information about DCE environment variables.

Chapter 2. Global and Cell Considerations

The purpose of "Chapter 2. Global and Cell Considerations" through "Chapter 5. Overview of DCE Maintenance" on page 33 is to assist you in planning for the installation, configuration, and maintenance of DCE. For more detailed information on installing and configuring DCE 2.2 for AIX, see the *IBM DCE for AIX, Version 2.2: Quick Beginnings*.

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 *IBM DCE for AIX, Version 2.2: Introduction to 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, 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. 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 connects cells with 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 the DNS directory services. The DNS name is the cell's alias.

  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 (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

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 DNS or CDS, you must obtain a globally unique name for your cell. The next sections describe how to establish DNS names for your cell. See the *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components* for a description of the valid characters supported in 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. To add a new name for your cell, use the **dcecp cellalias** task object.

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. 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 *IBM DCE for AIX, Version 2.2: 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.

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?

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.

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:

- Move server processes from the host.
- Unconfigure the host from the old cell, using the **unconfig.dfs** and **unconfig.dce** commands.
- Use the **config.dce** and **config.dfs** commands to reconfigure the host in the new cell.

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 the *IBM DCE for AIX, Version 2.2: Quick Beginnings*. The *IBM DCE for AIX, Version 2.2: Administration Guide—Core*

Components, the *OSF DCE GDS Administration Guide and Reference*, and the *IBM DCE for AIX, Version 2.2: DFS Administration Guide and Reference* provide details about the names that the DCE components use. The cell namespace can be divided into these 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 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 *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*. In Figure 2, **/.:** and **cell-profile** are well-known names.

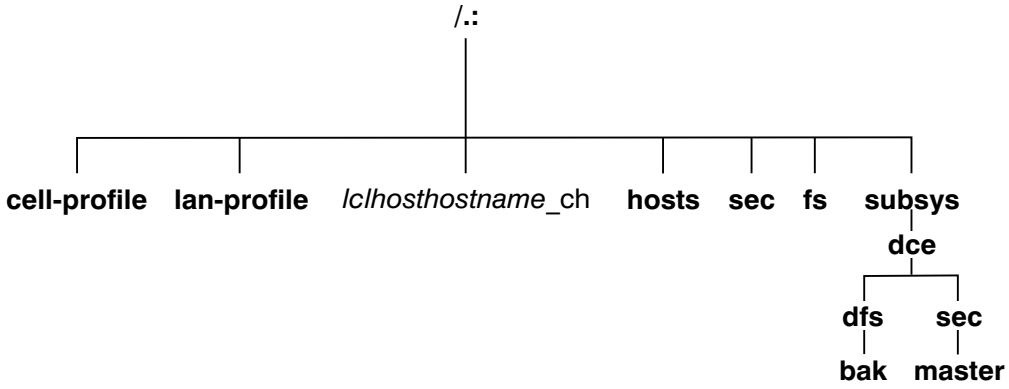

Figure 2. Top Level of the Cell Namespace

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.

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**.

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.

Note: Although DCE supports a principal name up to 1024 bytes long, AIX has a limit of 8 bytes.

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 four types of objects:

- Principals
- Groups
- Organizations
- Accounts

The *IBM DCE for AIX, Version 2.2: 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**
- **/.:/sec/xattrschemas**

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 *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*.

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 SMIT or 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 “The sec/group/subsys Directory” on page 78 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.

The Filespace

The following subsections contain guidelines for planning your cell’s filespace. The *IBM DCE for AIX, Version 2.2: 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.

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.

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.

Determining the Roles of DFS Machines

Follow the recommendations in the *IBM DCE for AIX, Version 2.2: 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 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. "Setting Up Filesets" on page 16 and the *IBM DCE for AIX, Version 2.2: 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.

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 *IBM DCE for AIX, Version 2.2: DFS Administration Guide and Reference* .)

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

- The **common** 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 contains files that users want to make available to everyone, including foreign and unauthenticated users.
- The **sys_type** 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 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 contains source filesets, such as those for DFS source files.

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 *IBM DCE for AIX, Version 2.2: 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.

Using @sys and @host Variables

Follow the suggested conventions in the *IBM DCE for AIX, Version 2.2: 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 *IBM DCE for AIX, Version 2.2: 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.

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.

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 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 by the **start.dce** command. The **start.dce** command can be invoked from **/etc/inittab** by specifying the **-autostart yes** option on the **config.dce** command or by adding **/etc/dce/rc.dce** to this file. This will allow DCE services to be brought up each time the machine boots. See the *IBM DCE for AIX, Version 2.2: Quick Beginnings* for information on the SMIT menu for starting DCE 2.2 for AIX at reboot.

- The DCE control program (**dcecp**) for the management and maintenance of the DCE RPC software. “DCE Administration Utilities” on page 25 describes **dcecp**. In addition, **rpccp** is used for operating on local registries.

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.

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.

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
  - Using the **–p** option, run as a proxy of the CDS Server so that the advertiser can forward server broadcasts into its own cell.
- 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.

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. “DCE Administration Utilities” on page 25 describes **dcecp**.

Slim Client Programs

In general, client systems are more likely to be memory constrained than server systems. When configuring the DCE software on a client system, more DCE daemons than might be necessary are started. If a client does not offer DCE services to other systems in the cell, it might not need all of the functions provided by these daemons. The DFS client and **dceunixd** can run with the Slim client.

Since no information about the Slim client is kept in the cell, **cell-admin** is not required to comfigure it. Instead use the **config.dce** command to configure the Slim client. Use the **start.dce** and **stop.dce** commands respectively to start and stop the Slim client. To unconfigure the Slim client use the **unconfig.dce** command.

The Slim client option reduces DCE memory consumption on client systems.

The Slim client runs a single instance of the CDS clerk with no other DCE daemons. Running a single instance of the CDS clerk is done by starting the clerk with the **-n** option. This starts a clerk without the CDS advertiser. However, if there are so many other DCE services and functions that can be run, how can a single CDS clerk be sufficient? The answer is that most DCE clients need only the following DCE functions:

- RPC calls (both authenticated and unauthenticated)
- DCE login
- CDS name lookups

For RPC calls and most logins, no DCE daemons are needed. These functions simply use RPC runtime routines and Security runtime routines.

For CDS name lookups, only a CDS clerk is necessary. With full DCE, CDS clerks are started by the CDS advertiser, requiring a CDS advertiser to be present. However, in DCE 2.2 for AIX, the **-n** option on the **cdsclerk** command starts a single instance of the CDS clerk without needing the advertiser. This clerk will not terminate after 20 minutes, as it does in full DCE. Additionally, when the clerk is started in this fashion, it takes over the role of the CDS advertiser in managing the CDS client cache.

Without an advertiser, the **cdsclerk** can not be managed by **dcecp** or **cdscp**. The following commands will fail:

```
cdscp show clerk
cdscp disable clerk
cdscp show cached clearinghouse
cdscp define cached server
cdscp show cached server
cdscp clear cached server

dcecp -c cdscache create
dcecp -c cdscache delete
dcecp -c cdscache show -server
dcecp -c cdscache show -clearinghouse
```

The services that compose **dced** and the functions that are disabled, because **dced** is not running on a DCE client system, are:

- **dced Endpoint Mapper Service** must run on any system providing a service that can be accessed through Remote Procedure Calls (RPCs). Such a server is called an RPC server. When a system issues an RPC to an RPC service, it uses the RPC runtime routines to send the request to a specific machine address and

asks for the desired RPC service by name. After the RPC reaches the machine where the service resides, the Endpoint Mapper Service maps the RPC service name to the endpoint, or port number, of the specific program providing the service. After the endpoint is known, the client is bound to the specific RPC service and RPCs can be issued directly to that service.

Although every DCE client system issues RPCs, most do not need the Endpoint Mapper Service, because they are probably not RPC servers. Therefore, the RPC-related limitation of not running **dced** on a client system is that it cannot be an RPC server.

- **Security Validation Service** provides the functions listed below. If a client system does not need these functions, it does not need the **dced** Security Validation Service. Note that a **dce_login** and authenticated RPCs can still be issued on a system that does not have this service running.
  - **Security Server Certification** ensures that the client is talking to a valid DCE security server during login. This is actually a server-type function, in that other DCE components issue RPCs to **dced** to use it.
  - **Third-party pre-authentication during dcel_ogin** In DCE 2.2 for AIX, the **dce_login** function assumes the user wants third-party pre-authentication, which requires the dced Security Validation Service. (This third-party pre-authentication is attempted before verifying that the user account is configured to require third-party pre-authentication.) However, if the Security Validation Service is not active during **dce_login**, **dce_login** avoids third-party pre-authentication, and does timestamp pre-authentication. No error is issued unless the user is configured at the server to require full third party pre-authentication.
  - **Keeping the machine context up to date**. Every machine that executes DCE has a principal name that it logs in as, and under which its DCE daemons run. This machine login is required by DCE Server daemons in order to acquire Privilege Ticket Granting Tickets (PTGTs) to access other DCE components. One role of the **dced** Security Validation Service is to ensure that this machine PTGT does not expire. This process is referred to as keeping the machine context up to date. Only DCE servers (when a server is something that receives RPCs) need the machine context kept up to date.
  - **Password and group overrides** One role of the security validation service is to process the overrides for DCE user and group information from the **passwd** and **group** files in **/opt/dcelocal/etc/**. No error is given if the administrator creates these files on the local machine. These files are ignored by the Slim client.
- **Security Integration** can run on a Slim client. Be aware that because the certification service is not available, when a user logs in, the user's identity cannot be certified to have been issued by a legitimate security server and that security integration on a Slim client cannot use psswd and group overrides. Because the machine context is not available, security integration on a slim client requires unauthenticated access to the registry.
- **System Management Services**:

  The system management functions provided by DCE are listed below. Without **dced**, a client system cannot be remotely managed by means of these functions.
  - **Host Data Management**: This service maintains local files of host data (that includes the host name, cell name, and cell aliases) and a post-processor file. The post-processor file contains program names that are matched to other host data items. **dced** runs the program if the corresponding host data item changes.

- **Server Control**: This service maintains data that describes the startup configuration and execution state for each server. It can also start or stop particular servers, and enable or disable specific services of servers. This service is not needed on a client that is not running any RPC servers.
- **Key Table Management**: This service allows for the remote maintenance of a server's key tables. This service is not needed on a client that is not running any RPC servers.

Because of limitations inherent in the Slim client, the following **fts** commands are disabled when issued on a machine configured as a DCE Slim Client.

fts create
fts clone
fts clonesys
fts delete
fts dump
fts move
fts release
fts restore
fts zap

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

Note: DFS will not recognize any DCE credentials acquired on a DCE client machine before the DFS client is configured. After you configure the DFS client, you must run the **dce_login** command to have authenticated access to files and directories in the DFS file space. Refreshing the credentials with the **kinit** command is *not* sufficient. The **dce_login_noexec** command does *not* authenticate the issuer to DFS. If your machine is configured to support AIX or DCE or both integrated security operations, you can use AIX commands like **login** or **su** to acquire DCE credentials that are recognized by DFS after the DFS client has been configured.

Requirements for DCE Server Machines

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

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 “Requirements for DCE Client Machines” on page 17 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.

DCE RPC Server Programs

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

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 registry, management and maintenance of the Security software. “DCE Administration Utilities” on page 25 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 the *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*.

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.

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. In addition, the **cdscp** program is used for controlling and displaying information about CDS clerks and servers. “DCE Administration Utilities” on page 25 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. See “Appendix A. Moving an Initial CDS Server” on page 47 for information on moving an initial CDS server.

In a DCE configuration that uses 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 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.

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 can be 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. “DCE Administration Utilities” on page 25 describes **dcecp**.
- If there are less than three time servers configured in the cell, one of the following command should be used:

  ```
  dtscp set servers required n
  (where n is the number of time servers in the cell)
  dcecp dts modify -miniservers n
  (where n is the number of time servers in the cell)
  ```

  This will prevent a warning message from being logged every time the server attempts to synchronize.

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 the *IBM DCE for AIX, Version 2.2: 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 *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*also explains the criteria you need to use when selecting a time source for your network to use.

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
- DFS Backup Server machine
- DFS Fileset Replication Server machine

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.

DCE Control Program

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

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

DCE RPC Administration Programs

The **rpccp** program, which is the RPC-specific administrative tool, allows you to browse, update, add, and delete the DCE RPC attributes of entries stored in the CDS namespace and the endpoints that are managed by local and remote **dced** daemons.

Security Service Administration Programs

The Security Service provides the following administration utilities:

- The **dcecp acl** command displays, adds, modifies, and deletes ACL entries for a specific object. The *IBM DCE for AIX, Version 2.2: Administration Commands Reference* contains detailed information about using the **dcecp acl** command.
- The **dcecp account**, **group**, **organization**, **principal**, **registry**, **user**, and **xattraschema** commands allow you to edit the registry database or the local registry. Almost all editing of the registry database must be done with these commands. The *IBM DCE for AIX, Version 2.2: Administration Commands Reference* explains the use of the commands.
- The **passwd_import** command allows you to 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** and **group_override** files allow you to establish overrides to the information contained in the registry.
- The **rmxcred** command purges expired tickets from the credentials directory.

- The **dcecp registry** command helps you manage server replicas of the registry, change the master server site, and reinitialize a slave server. This command also helps you manage the security server and its database. You can perform tasks such as generating a new master key for the database and stopping the security server.

CDS Administration Programs

CDS provides the following administration utilities:

- The CDS control program, **cdscp**, is a command interface used to control CDS servers and clerks and manage the namespace and its contents. The **cdscp** command interface was available with previous versions of DCE and is provided to ease migration to the use of the **dcecp** utility. For more information about the CDS control program, see the *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*.
- The **cdsli** gives a DCE user the ability to recursively list the namespace of cells.
- The **cdsdel** deletes recursively the namespace of cells.
- The CDS Browser, **cdsbrowse**r, is a program based on Motif that lets you view the contents and structure of a namespace.
- The CDS Advertiser, **cdsadv**, allows applications to access and communicate with **cdsd**, starts any needed CDS clerks and creates the cache shared by local CDS clerks.
- The CDS clerk, **cdsclerk**, is an interface between CDS client applications and CDS servers.
- The DCE control program, **dcecp**, can be used to browse, update, and delete CDS entries, and to manage the namespace. It replaces **cdscp**.

  The **mkreg.dce** command enters information about your DCE cell into the database maintained by your domain name server (the **named** daemon).

  The **rmreg.dce** command removes information from the database maintained by your domain name server (the **named** daemon) that were added by the **mkreg.dce** command.

DTS Administration Programs

The DTS control program (**dtscp**) allows you to administer DTS, including configuring the **dtsd** daemon as either a client or a server. The **dtscp** program is included in the DCE client software.

Programs for DCE Remote Administration Machines

A system user can perform administrative tasks from any machine in the cell when logged in as the cell administrator. In order to perform administrative tasks, the user must authenticate and ensure that the tools necessary to perform the tasks are available. The administration utilities that need to be installed, in addition to the DCE client software, are the following:

- The **cdscp** program for controlling certain operations on CDS clerks and servers.
- The **rgy_edit** program for maintaining local copies of the Registry.
- The **passwd_override**, **sec_admin**, **sec_create_db**, and **sec_salvage_db** programs must be installed for Security service administration.
- The **bak**, **bos**, **cm**, **dfsexport**, **fts**, **newaggr**, **salvage**, **scout** programs for DFS administration.

- The **cdscp** program for controlling certain operations on CDS clerks and servers.
- The **rgy_edit** program for maintaining local copies of the Registry.

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

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

The files used by DCE are grouped in the following locations:

- The **dcelocal** subdirectories
- Conventional UNIX subdirectories.

Some information needs to be kept locally on a machine for reliability and to ensure 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.

The **dcelocal** subtree is created when you install DCE components.

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**.

This chapter contains the following topics:

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 installed during the default DCE installation procedure be kept locally on server machines for stand-alone 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 each machine can run as a stand-alone 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 below **dcelocal/etc** (static configuration data) and **dcelocal/var/adm**. All server-specific data files are located in the **dcelocal/var/***dce-component-name* directory.

The default path for **dcelocal** is set to **dcelocal** during the configuration process. This is a fixed path name. Every machine must have local access to the files that are necessary to configure it (up to activating DFS access in the cell). The **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.

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

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**.

File Locations

The installation process for DCE 2.2 for AIX places files in the following locations:

/usr/lpp/dce
All DCE files except those in the remainder of this list.

/usr/lpp/dce/tcl/dcedcf
DCE/DFS configuration scripts

/usr/lpp/dcedoc
All DCE for AIX documentation files and their related tools.

/etc/dce
The following files:

- **rc.dce**
- **dce.clean**
- **rc.dfs**
- **dfs.clean**
- **rpc.clean**
- **rcnfs.dfs**

/etc/dce/rspfiles
Configuration response files

/etc/zoneinfo
Timezone rules for DTS.

/tmp/dce
Temporary location for configuration processing

/usr/lib/nls/msg/en_US
English message catalogs.

/usr/include
Include files (mostly under **/usr/include/dce**)

/usr/lib
libdce.a, **libcfgdce.a**, **libdcelibc_r.a**, **libdcepthreads.a**, and **libidlcxx.a**

/usr/lib/security
The **DCE** load module for AIX/DCE integrated security operations.

/usr/lib/drivers
DFS kernel extensions.

/usr/sbin
Commands for loading the DFS kernel extensions.

dcelocal/ is set up as a symbolic link to **/usr/lpp/dce**. **dcelocal/var** is set up as a symbolic link to **/var/dce**. **dcelocal/etc** is set up as a symbolic link to **/etc/dce**. **dcelocal/tmp** is a symbolic link to **/tmp/dce**. A link for each of the DCE commands is placed in **/usr/bin**.

In addition, SMIT objects are loaded into the Object Data Manager (ODM) database.

Files Created at Runtime

The following are files created at runtime:

- **cfg.dat** contains data about the current configuration state of the DCE/DFS components
- **cfgdce.dat** contains non-viewable configuration data
- **cfgdce.log** configuration log
- **cfgdce.bck** backup of the cfgdce.log - created when the cfgdce.log file exceeds 100000 bytes
- **protseqs.rpc** lists the protocol being used.
- **dce_cf.db** contains the cell name and the DCE host name of the local machine

File Systems to Create and Mount

You will probably want to create new AIX JFS file systems in order to use DCE effectively:

/var/dce

All DCE components store information in the **/var/dce** directory. If the **/var** file system fills up, DCE and other subsystems that depend on **/var** (such as the mail and spooler sybsystems) cannot operate correctly.

You should create a new file system mounted over **/var/dce** before you install DCE. You should reserve about 30 megabytes for **/var/dce** for your initial DCE configuration.

/var/dce/directory

Within this directory tree is where the CDS server stores the clearinghouse files, which contain this server's portion of the namespace, and local data.

If this machine is configured as a CDS server, it is recommended that you create a new file system mounted over **/var/dce/directory** before you install DCE.

You should reserve about 30 megabytes for the server's use.

If you do not plan to create a separate files system for the CDS server, you should add the additional 30 megabytes to **/var/dce**.

/var/dce/security

This is where the Security Server stores the registry, credentials, and local data. If this machine will be a Security Server, you should add an additional 10 megabytes to **/var/dce** for the server's use.

DFS client cache

If the machine will be configured as a DFS client and you plan to configure DFS to use on-disk caching, you should create a new file system to hold the DFS cache files. This must be an AIX JFS filesystem which is not using compression or fragmentation. The default directory is

/var/dce/adm/dfs/cache, but you can specify a different directory when you configure DFS. The default cache size is 10MB, but you can change this during configuration. If you do not want to create a separate file system, ensure that the file system where you plan to place the DFS cache has enough room to hold the cache or configure the DFS cache to be in-memory.

If you do not plan to create a separate file system for the DFS cache directory and you use the default cache directory, you should add room for the DFS cache to **/var/dce**.

Files stored in **/var/dce** are any files particular to the individual machine. You should monitor the space usage in **/var/dce** (and any associated separate files systems) to make sure it does not fill up. To clean up expired credentials files in **/var/dce**, use the **/usr/lpp/dce/bin/rmxcred** command. The DCE Auditing facility also uses space in **/var/dce**. See the *IBM DCE for AIX, Version 2.2: Administration Commands Reference* for more information on **rmxcred** and DCE Auditing.

DCE Daemon Core Locations

The following list shows the locations of DCE daemons and where they dump core.

- **secd** - **/var/dce/security/adm/secd**
- **dced** - **/var/dce/dced**
- **cdsd** - **/var/dce/directory/cds/adm/cdsd**
- **gdad** - **/var/dce/directory/cds/adm/gdad**
- **csdadv** - **/var/dce/adm/directory/cds/cdsadv**
- **cdsclerk** - **/var/dce/adm/directory/cds/cdsclerk**
- **dtsd** - **/var/dce/time/adm/dtsd**
- **dceunixd** - **/var/dce/security/adm/dceunixd**

Also, if you are using DFS, you will have the following:

- **dfsbind** - **/var/dce/dfs/adm/dfsbind**

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 *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*, *IBM DCE for AIX, Version 2.2: DFS Administration Guide and Reference*, and *OSF DCE GDS Administration Guide and Reference*.

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.

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 the *IBM DCE for AIX, Version 2.2: 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. The *IBM DCE for AIX, Version 2.2: 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 *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components* .

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 the *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*.

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.

The *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components* provides detailed information about how to perform these tasks.

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.

The *IBM DCE for AIX, Version 2.2: 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.

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 *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components* .

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 *IBM DCE for AIX, Version 2.2: 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.

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. However, neither the example code nor the external time providers are supported by IBM.

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 *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*.

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. The *IBM DCE for AIX, Version 2.2: 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 the *IBM DCE for AIX, Version 2.2: 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 *IBM DCE for AIX, Version 2.2: 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 the *IBM DCE for AIX, Version 2.2: 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 the *IBM DCE for AIX, Version 2.2: Administration Guide—Core Components*.

  Note: If the cell policy for password expiration or password lifetime is set to something other than **forever**, it applies to all principals, including **dce_rgy**. If you exceed this limit, **secd** will not be able to authenticate. Thus, your security server will not be operational. It is recommended that you set these limits to **forever** and set other limits on the basis of organizations or accounts.

- Backing up the registry

  The *IBM DCE for AIX, Version 2.2: 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 *IBM DCE for AIX, Version 2.2: Administration Commands Reference* provides descriptions of audit-related **dcecp** objects and commands. See the *IBM DCE for AIX, Version 2.2: 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. The *IBM DCE for AIX, Version 2.2: 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

Reconfiguring the Registry

There are two main reconfiguration tasks included in the administration of the Security Service. The following tasks are described in the *IBM DCE for AIX, Version 2.2: 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

Removing Expired Credentials Files

The **/var** file system may periodically fill up, especially if expired credentials files are not removed. If you are root, you can use the **rmxcred** tool to remove expired tickets from the credentials directory (**dcelocal/var/security/creds**). You should invoke this tool on a regular basis, such as setting up an AIX **cron** job to run the tool daily.

Part 2. Additional Configuration Information

Chapter 6. Configuration Response Files

A response file contains the information that you would normally specify on the command line. Using a response file allows you to automate your configuration process by eliminating the need to enter the information on the command line. Response files can be used with the **config.dce** and **unconfig.dce** commands. For more information about these commands see the *IBM DCE for AIX, Version 2.2: Administration Commands Reference* .

If you prefer to automate only part of the process, you can create a partial response file that contains information for only one option or a few options. You can then specify the remaining options on the command line.

Each line in the response file contains a keyword and an associated value. The value is used by the configuration program as if it were input on the command line. Consider this example:

```
dce_hostname=georgia
```

In this example, the keyword is **dce_hostname** and the associated value is **georgia**. The configuration program uses georgia as the value for the **-dce_hostname** command line option.

The DCE for AIX package includes one sample response file, **DCE Client Services**. Use this file to create your own response files. Perform the following:

1. Copy /opt/dcelocal/etc/rspfiles/dce_smpl.rsp as *XXXXXXXX*.RSP, where *XXXXXXXX* is a name that you choose for your new response file.
2. Edit the file by changing the keyword values to fit your installation.

   Note: You can remove keywords and change the values, but do not change the keyword names.

Key words for DCE Response Files

The DCE configuration response file contains two types of information: cell information and host (machine) information. (Typically, cell information is stored in one response file and host information is stored in another response file. The files are then linked together by including one of the files in the other file. This allows the machines in a cell to each use the same cell response file and an individual host file.)

All information in a response file is optional. If the DCE configuration program requires a piece of information that is not included in the response file. the configuration interface informs you that the information is missing.

You can create response files using an ASCII text editor. When you create a response file with an editor, refer to the following formatting rules:

- Every line is either a comment or a response line. Blank lines are treated as comment lines.
- The first nonblank charaacter of a comment line must be and asterisk (*) or a number sign (#). Comments cannot be imbedded in response lines.
- Keywords are not case-sensitive and can begin anywhere on the line. However, a keyword with a single value must be contained entirely on one line.
- Response lines contain information in one of the following formats:

Value string

The response line contains the keyword, an equals sign (=), and a value. For example,

```
config_type=local
```

A value string must be contained on one line and cannot contain parentheses.

Value list

The response line contains the keyword, the equals sign(=), and several values contained in parentheses and separated by commas. Each value in the value list can be a single value or a value string. For example,

```
host_id=(
  tcpip_name=chariot.roma.italia.com
)
```

A value list can also contain other value lists, or it can be empty. The end parenthesis ″)″ on a line by itself marks the end of the value list.

- Keywords within a section of a response file can be in any order.
- To imbed a response file into another response file, use the **include** keyword. For example:

```
⋮
  include /mydir/svrresp2.rsp
⋮
```

Cell Section Keywords

The cell section of a response file contains keywords and information that apply to the entire cell. For this reason, you may want to create a response file for the cell information only. You can then include this file in several host response files.

The following keywords specify information associated with the cell. Use these keywords in the cell section only.

Table 1. DCE Cell Keywords

KEYWORD	VALUE TYPE	VALUE RESTRICITONS	COMMAND LINE OPTION
cell_name	PCS (except a space and the @, **:**, and **-** characters)	Maximum length: 399	**-cell_name**
cell_administrator	PCS (except a space and the @, **:**, and **-** characters)	Maximum length: 255	**-cell_admin**
max_unix_id	Numeric	Range: 0-2, 147, 483, 647	**-max_unix_id**
min_group_unix_id	Numeric	Range: 0-2, 147, 483, 647	**-min_group_id**
min_organization_unix_id	Numeric	Range: 0-2, 147, 483, 647	**-min_org_id**
min_principal_unix_id	Numeric	Range: 0-2, 147, 483, 647	**-min_princ_id**

Table 1. DCE Cell Keywords (continued)

KEYWORD	VALUE TYPE	VALUE RESTRICITONS	COMMAND LINE OPTION
master_security_server	Value list (see Table 3 on page 44)	Maximum length: 255	**-sec_master**
cds_server	Value list (see Table 3 on page 44)	Maximum length: 255	**-cds_server**

Host Section Keywords

Host informtion is contained within a host section in a response file. A host section begins with a **host=(** keyword and ends with a right parenthesis ″**)**″. The host section of a response file consists of host keywords only; any cell keyword within a host section is ignored.

There are two types of host sections:

Global
A response file can contain only one global host section, which is for information that is used to initialize configuration values for subsequent specific host sections. The global section of a response file does not contain the **dce_hostname** or **host_id** keywords. If a global host section is included in a response file, it must be the first host cection encountered. If the DCE Configuration program encounters a gloval host section after the first host section, a worning is written to the log file and the global section is ignored.

Specific
Contains the **dce_hostname** or **host_id** keywords. A specific host section contains configuration values for a particular host. The specific host sections override global values by including the keywords and setting them to different values. A response file can contain multiple specific host sections

The following table describes the host information keywords:

Table 2. DCE Host Keywords

KEYWORD	VALUE TYPE	VALUE RESTRICITIONS	COMMAND LINE OPTION
host	Value list	Table 3 on page 44	(not applicable)
config_type	Option	Choices: **full**, **admin**, **local**	**-config_type**
dce_hostname	Text	Maximum length: 255	**-dce_hostname**
lan_profile	Text	Maximum length: 255	**-lan_profile**
host_id	Value list	Table 3 on page 44	(not applicable)
protocols[1]	Text	Choices: **tcpip_connection_oriented**, **tcpip_connectionless**, **all**, **none**	**-protocol**
autostart	Option	Choices: **yes**, **no**	**-autostart**
sync_clock	Option	Choices: **yes**, **no**	**-sync_clocks**
sync_server	Value list	Table 3 on page 44	**-time_server**

Table 2. DCE Host Keywords (continued)

KEYWORD	VALUE TYPE	VALUE RESTRICITIONS	COMMAND LINE OPTION
components	Value list	Table 4	(not applicable)
Note: 1. This keyword has comma-separated values. There cannot be any spaces between the commas and the tokens. For example: `protocols=tcpip_connection_oriented,tcpip_connectionless`			

Keywords for Identifying Machines

Several cell and host keywords require a value list that contains the network ID of a machine (usually a server) that the host communicates with. The network ID for a machine can be specified using one or more of the following keywords with a value.

Table 3. Host Identification Value

KEYWORD	VALUE TYPE	VALUE RESTRICITIONS	COMMAND LINE OPTION
tcpip_name	IP host name	Dotted format, maximum length: 255; example: `uniqienm.domain.ibm.com`	**-host_id**
tcpip_addr	IP address	Numeric separated by periods, maximum length: 15; example: `123.456.789.109`	**-host_id**

Values for the Components Keyword

The components keyword identifies the components to configure on a host. The components value list can contain several of the components listed in Table 4.

The following table lists the possible keywords you can use to specify components and the values that each component requires:

Table 4. Components Keywords

KEYWORD	DESCRIPTION	VALUE TYPE	VALUE	COMMAND LINE OPTION
client	All client components	Value list	Table 5 on page 45	**all_cl**
slim_client	The slim client component	Value list	Table 5 on page 45	**slim_cl**
sec_cl	Security client	Value list	Table 5 on page 45	**sec_cl**
sec_svr	Master Security server	Value list	Table 5 on page 45	**sec_srv**

Table 4. Components Keywords (continued)

KEYWORD	DESCRIPTION	VALUE TYPE	VALUE	COMMAND LINE OPTION
sec_rep	Security replica sever	Value list	Table 5	**sec_rep**
cds_cl	CDS client	Value list	Table 5	**cds_cl**
cds_svr	Initial CDS server	Value list	Table 5	**cds_srv**
cds_second	Additional DCS server	Value list	Table 5	**cds_second**
gda_svr	Global Directory Agent	Value list	Table 5	**gda**
dts_client	DTS client	Value list	Table 5	**dts_cl**
dts_local	DTS local server	Value list	Table 5	**dts_local**
dts_global	DTS global server	Value list	Table 5	**dts_global**
rpc	RPC	Value list	Table 5	**rpc**
sec_audit	audit	Value list	Table 5	**audit**
pw_strength_svr	Password synchronization server (multiples are permitted)	Value list	Table 5	**pw_strength_srv**
ems	Event Management Service	Value list	Table 5	**ems_srv**
snmp	DCE SNMP Subagent	Value list	Table 5	**snmp_srv**
dceunixd	Integrated Login	Value list	Table 5	**dce_unixd**

Table 5. Components General Keywords

KEYWORD	VALUE TYPE	VALUE RESTRICIONS	COMPONENTS	COMMAND LINE OPTION
config_state	Option	Choices: **configured**, **not_configured**	All components	(not applicable)
unconfig_depend	Option	Choices: **yes**, **no**	All components	**-dependents**
force_unconfig	Option	Choices: **yes**, **no**	All components	**-force**
name	Text	Maximum length 255	secrep, secsvr	**-sec_server_name**
proxy	Option	Choices: **yes**, **no**	cds_cl	**-proxy**
clearinghouse	Text	Maximum length 255	cds_second	**-clr_house**

Table 5. Components General Keywords (continued)

KEYWORD	VALUE TYPE	VALUE RESTRICIONS	COMPONENTS	COMMAND LINE OPTION
ldap_server	Text	Maximum length 255	gda_svr	**-ldap_server**
role	Option	Choices: **noncourier**, **courier**, **backupcourier**	dts_local, dts_global	**-courier_role**
server_command	Text	Maximum length 255	pw_strength_svr	**-pwdstr_cmd**
command_args	Text	Maximum length 255	pw_strength_svr	**-pwdstr_arg**
account	Value list	Table 6	pw_strength_svr	(not applicable)
no_pesite_update	Option		rpc	**-no_pesite_update**
pesite_update_time	Numeric	Range: 100-1440	rpc	**-pesite_update_time**
snmp_trap	Text		snmp	**-snmp_trap**
num_dce_unixd	Numeric	Range: 1-5	dceunixd	**-num_dce_unixd**
cache_lifetime	Numeric	Range: 2-120	dceunixd	**-cache_lifetime**

Table 6. Account Keywords

KEYWORD	VALUE TYPE	VALUE RESTRICTIONS	COMMAND LINE OPTION
name	Text	Maximum length: 255	**-pwdstr_principal**
protect_level	Option	Choices: **pktprivacy**, **pktinteg**, **cdmf**	**-pwdstr_protect_level**

Appendix A. Moving an Initial CDS Server

Note: This procedure only works if no second CDS server has already been defined. It will not work if one or more CDS servers are active and master replicas are spread over different clearinghouses. When defining the replica set, you need to specify all existing replicas.

1. Create an additional CDS server on the machine that you want to become the initial CDS server.

   ```
   # make cds_second
   or
   # smit mkcdssrv
   ```

2. Log in to DCE as cell_admin on both machines.
3. Use the **cdsli** command to verify the directories in CDS. Copy all directories from BoxA to BoxB.

   ```
   # cdsli -dR
   ```

4. Verify that the additional CDS server is empty on BoxB.

   ```
   # cdscp
   cdscp> set cdscp preferred clearinghouse /.:/<BoxB>_ch
   cdscp> show dir /.:/*
   cdscp> quit
   ```

5. Replicate all CDS directories to the new additional CDS server's (BoxB) clearinghouse, and make them the master replicas:

   ```
   # for dir in $ (cdsli -R); do
   > echo "Creating replica for $dir"
   > cdscp create replica $dir clear /.:/<BoxB>_ch
   > done
   ```

   ```
   # for dir in $ (cdsli -R); do
   > echo "Swapping master CDS attribute for $dir"
   > cdscp set dir $dir to new epoch master /.:/<BoxB>_ch readonly /.:/<BoxA>_ch
   > done
   ```

   ```
   cdscp set dir /.: to new epoch master /.:/<BoxB>_ch readonly /.:/<BoxA>_ch
   ```

6. Verify that the swap worked. You should see that the master replica for everything, including the /.: directory is located on BoxB's clearinghouse:

   ```
   # cdscp
   cdscp> show dir /.:
   cdscp> show dir /.:/*
   ```

7. Stop and restart DCE to delete the CDS cache on both machines:

   ```
   # stop.dce
   # cd /var/dce/adm/directory/cds
   # rm cds_cache.*
   # start.dce
   ```

8. If you want to remove the additional CDS server from BoxA. Ensure that you are logged into DCE as cell_admin.

   ```
   # rmdce cds_second
   ```

9. Start and restart DCE to perform a CDS client cache refresh on all systems in the cell.

   ```
   # stop.dce
   # cd /var/dce/adm/directory/cds
   # rm cds_cache.*#
    start.dce
   ```

Appendix B. Environment Variables

Environment variables are variables used by DCE that customers can set themselves. These variables are described in the following sections on Audit, CDS/XDS, Configuration, IDL, NLS/Security, RPC, and Security variables.

Audit Variables

Setting the DCE Audit Environment variables is discussed in the following topics.

DCEAUDITON

Purpose

Turns auditing on for an application.

Synopsis

```
DCEAUDITON = <any_value>
```

Description

If this variable is declared at the time the application is started, auditing is turned on.

The presence or absence of this variable at start time can be used to select which applications use auditing.

Examples

```
DCEAUDITON=1
```

DCEAUDITOFF

Purpose

Turns auditing off for an application.

Synopsis

```
DCEAUDITOFF = <any_value>
```

Description

If this variable is declared at the time the application is started, auditing is turned off. This takes precedence over **DCEAUDITON**. If both are declared, auditing is turned off. Auditing is off by default.

The presence or absence of this variable at start time can be used to select which applications use auditing.

Examples

```
DCEAUDITOFF=1
```

DCEAUDITFILTERON

Purpose

Turns event filtering on for an application.

Synopsis

```
DCEAUDITFILTERON = <any_value>
```

Description

If this variable is declared at the time the application is started, audit filtering is turned on. It is off by default.

The presence or absence of this variable at start time can be used to select which applications use event filtering.

Note: If filtering is turned on in a program that does not export its bindings to the endpoint map (for example, not a RPC-based application server), auditing will fail to process any events generated by that program.

Examples

```
DCEAUDITFILTERON
```

DCEAUDITTRAILSIZE

Purpose

Sets the maximum size of an audit trail.

Synopsis

```
DCEAUDITTRAILSIZE = <size_in_bytes>
```

Description

If this variable is declared at the time the application is started, its value specifies the maximum size of the audit trails to which it writes.

Examples

```
DCEAUDITTRAILWRAP=1024
```

DCEAUDITTRAILWRAP

Purpose

Sets the storage strategy for the central audit trail.

Synopsis

```
DCEAUDITTRAILWRAP = <any_value>
```

Description

If this variable is declared at the time the audit daemon is started, the central audit trail will use the wrap storage strategy (**aud_c_trl_ss_wrap**). When wrapping is turned on, audit starts writing audit records until it reaches the size limit. Then it wraps around to the beginning of the trail and continues writing audit records from there. The save storage strategy (**aud_c_trl_as_save**) is used by default.

Examples

```
DCEAUDITTRAILWRAP=1
```

DCEAUDITWRAP

Purpose

Sets the storage strategy for any application to wrap.

Synopsis

```
DCEAUDITWRAP = <any_value>
```

Description

If this variable is declared at the time the application is started, the audit trail to which the application writes will use the wrap storage strategy (**aud_c_trl_ss_wrap**). When wrapping is turned on, audit starts writing audit records until it reaches the size limit. Then it wraps around to the beginning of the trail and continues writing

audit records from there. The save storage strategy (**aud_c_trl_ss_store**) is the default. This only applies to audit trails to which the application sends audit records.

Examples

```
DCEAUDITWRAP=1
```

SECDAUDITWRAP

Purpose

Sets the storage strategy for the security server to wrap.

Synopsis

```
SECDAUDITWRAP = <any_value>
```

Description

If this variable is declared at the time the security server is started, it will use the wrap storage strategy (**aud_c_trl_ss_wrap**). When wrapping is turned on, audit starts writing audit records until it reaches the size limit. Then it wraps around to the beginning of the trail and continues writing audit records from there. The save storage strategy (**aud_c_trl_ss_save**) is the default.

Examples

```
SECDAUDITWRAP = 1
```

CONFIGURATION

The only DCE Configuration Environment variable is *dcelocal*. This variable equates to the AIX value set for the **/opt/dcelocal** path.

IDL

Setting the DCE IDL Environment variables is discussed in the following topics.

IDL_GEN_AUX_FILES

Purpose

Generate dummy auxiliary files for compilation purposes.

Synopsis

```
export IDL_GEN_AUX_FILES=<any value>
```

Description

The auxiliary files (**_caux.c** and **_saux.c**) generated by the IDL Compiler in older releases are no longer being generated. Only the stub and header files are needed. Older application Makefiles may contain references to these files. Users who migrate from an old release of DCE may need to remove references to these auxiliary files from their application Makefiles. Alternatively, the environment variable **IDL_GEN_AUX_FILES** can be set (to any value) to generate dummy auxiliary files to avoid Makefile errors.

Examples

```
export IDL_GEN_AUX_FILES=1
```

IDL_GEN_INTF_DATA

Purpose

Add storage information list to the type vector definition of the generated stub files.

Synopsis

```
export IDL_GEN_INTF_DATA=any value
```

Description

If the environment variable is not NULL, the IDL Compiler will add the storage information list to the type vector definition within the client stub and server stub files.

Examples

```
export IDL_GEN_INTF_DATA=1
```

NLS/SECURITY

Setting the DCE NLS/Security Environment variables is discussed in the following topics.

DCE_USE_NONPORTABLE_NAMES

Purpose

Extends OSF naming rules for PGO names to allow characters outside of the DCE portable character set.

Synopsis

```
DCE_USE_NONPORTABLE_NAMES=1
```

Description

According to standard (OSF) DCE, entries in the Security namespace, such as principal names, can be composed only of characters in the DCE portable character set (see the Architectural Overview of DCE in the Introduction to DCE). DCE 2.2 for AIX provides an override capability which enables the use of non-portable characters. This capability should be used only in environments that are homogeneous with respect to code set and only in environments in which all DCE installations support this extension. Security namespace entries that use non-portable characters are guaranteed to work correctly only when the code set of the entire enterprise is the same as that of the process under which the names are created. To enable non-portable Security names, this environment variable must be set before DCE is started, in all client and server processes in which DCE Security will run.

Examples

```
DCE_USE_NONPORTABLE_NAMES=1
```

DCE_USE_WCHAR_NAMES

Purpose

To improve performance in certain, user-restricted, Asian environments when processing CDS names.

Synopsis

```
DCE_USE_WCHAR_NAMES=0
```

Description

According to standard (OSF) DCE, certain entries in the CDS

namespace, such as directory names, can be composed of characters from outside of the DCE portable character set. Because DCE does not perform code set conversion on CDS names, non-portable characters should be used only in environments which are, and will remain, homogeneous with respect to the code set. If you are using an Asian locale, but you are restricting names to the portable character set, Directory performance can be improved by setting this environment variable to 0. By default, it is set to 1.

Examples

```
DCE_USE_WCHAR_NAMES=0
```

RPC

Setting the DCE RPC Environment variables is discussed in the following topics.

DCERPCCHARTRANS

Purpose

Point to a file containing replacement ASCII/EBCDIC translation tables.

Synopsis

```
export DCERPCCHARTRANS=file name
```

Description

The file named in the environment variable **DCERPCCHARTRANS** contains the replacement translation tables for the ASCII to EBCDIC and EBCDIC to ASCII tables. This file replaces the default table contained within the RPC runtime.

Examples

```
export DCERPCCHARTRANS=tmp.tab
```

RPC_CN_AUTH_SUBTYPE

Purpose

Determines the checksum algorithm used when RPC encodes a packet.

Synopsis

```
export RPC_CN_AUTH_SUBTYPE=number
     /* where number is 0 for DES and 1 for MD5  */
```

Description

This environment variable determines the checksum algorithm to be used when RPC encodes a packet. The acceptable values are 0 for an 8 byte DES checksum, or 1 for a 16 byte MD5 checksum. If the variable is not set, the default algorithm is MD5.

Examples

```
export RPC_CN_AUTH_SUBTYPE=0
```

RPC_DEFAULT_ENTRY

Purpose

Specifies the starting point for directory service searches.

Synopsis

```
export RPC_DEFAULT_ENTRY=entry name in the namespace
```

Description

Designates the default entry in the name service database that the import and lookup routines use as the starting point to search for binding information for a compatible server. Normally, the starting entry is a profile.

An application that uses a default entry name must define this environment variable. The RPC runtime does not provide a default. In particular, the environment variable is required when using the **auto_handle** IDL attribute.

For example, suppose that a client application needs to search the name service database for a server binding handle. The application can use the **rpc_ns_binding_import_begin** routine as part of the search. If so, the application must specify, to the routine's *entry_name* parameter, the name of the entry in the name service database at which to begin the search. If the search is to begin at the entry that the **RPC_DEFAULT_ENTRY** environment variable specifies, then the application must specify the value NULL to parameter *entry_name* in routine **rpc_ns_binding_import_begin**.

Examples

```
export RPC_DEFAULT_ENTRY=/.:/Servers
```

RPC_DEFAULT_ENTRY_SYNTAX

Purpose

Specifies the syntax of directory service entries.

Synopsis

```
export RPC_DEFAULT_ENTRY_SYNTAX=value
```

Description

Specifies the syntax for the name provided in the **RPC_DEFAULT_ENTRY** environment variable. In addition, it provides syntax for those RPC NSI routines that allow a default value for the name syntax parameter. Valid values are 0 for default syntax and 3 for DCE syntax. If the **RPC_DEFAULT_ENTRY_SYNTAX** environment variable is not defined, the RPC runtime defaults to the DCE name syntax.

Examples

```
export RPC_DEFAULT_ENTRY=3
```

RPC_DISABLE_EP_RESOLVE_V4

Purpose

Disables support for OS/390 load balancing.

Synopsis

```
export RPC_DISABLE_EP_RESOLVE_V4=any value
```

Description

This environment variable causes the RPC runtime to use version 3 of the **rpc_ep_resolve_binding** interface instead of version 4. Version 4 returns both an endpoint and an IP address, allowing OS/390 to return a different IP address for load balancing purposes. Version 3 returns only an endpoint.

Examples

```
export RPC_DISABLE_EP_RESOLVE_V4=YES
```

Purpose

Disables single-threaded behavior.

Synopsis

```
export RPC_DISABLE_SINGLE_THREAD=any value
```

Description

This environment variable is used to disable single threaded behavior in the client side of connectionless RPC applications. There should normally be no reason to use this environment variable.

Examples

```
export RPC_DISABLE_SINGLE_THREAD=YES
```

RPC_ITIMER_SIGNAL

Purpose

Specifies the type of itimer signal used when RPC is single threaded.

Synopsis

```
export RPC_TIMER_SIGNAL=SIGVTALRM or SIGALRM
```

Description

This environment variable is used to set the type of interval timer used when RPC is single threaded. The acceptable values are **SIGVTALRM** and **SIGALRM**. The default is **SIGVTALRM**. **SIGVTALRM** specifies a timer of type **ITIMER_REAL** and **SIGVTALRM** signals. **SIGALRM** specifies a timer of type **ITIMER_REAL** and **SIGALRM** signals.

Examples

```
export RPC_TIMER_SIGNAL=SIGVTALRM
```

RPC_MAX_UDP_PACKET_SIZE

Purpose

Sets the maximum UDP packet size.

Synopsis

```
export RPC_MAX_UDP_PACKET_SIZE=number
```

Description

The RPC runtime by default will break large RPC calls into 4352 byte UDP packets if the **ncadg_ip_udp** protocol is used. If larger packets should be supported, the **RPC_MAX_UDP_PACKET_SIZE** environment variable can be set to the largest size desired. This environment variable can also be set lower to prevent IP fragmentation of the UDP packets, which may be necessary if the packets are traversing a network with extremely limited resources or a firewall that is misconfigured and dropping fragments.

Examples

```
export RPC_MAX_UDP_PACKET_SIZE=16384
```

RPC_RESTRICTED_PORTS

Purpose

Restricts TCPIP port numbers used by RPC to a certain range.

Synopsis

```
export RPC_RESTRICTED_PORTS=1stprotseq[port#-port#]:2ndprotseq[port#-port#]
```

Description

This environment variable restricts the TCPIP port numbers used by RPC to a certain range. The problem is that RPC applications such as DFS will not work between sites which use router filtering as a security measure. These filters restrict incoming network packets to specific addresses on specific ports. Since RPC dynamically determines port numbers for its services it will not work in this environment. When this environment variable is used, RPC will only use port numbers in the specified ranges. Then the filters can be opened up over those ranges.

Examples

```
export RPC_RESTRICTED_PORTS=ncadg_ip_udp[5000-5500]:ncacn_ip_tcp[6000-6500]
```

```
export RPC_RESTRICTED_PORTS=ncacn_ip_tcp[5000-5500,5800-6000]
```

RPC_SUPPORTED_PROTSEQS

Purpose

Limits the protocol sequences used by the RPC runtime.

Synopsis

```
export RPC_SUPPORTED_PROTSEQS=protseq:protseq
```

Description

This environment variable is used to tell the RPC runtime to limit the set of supported protocol sequences to those specified. The syntax is to list the desired protocol sequence strings, separated by colons (not semicolons). The default is to use all protocol sequences, for example, **ncacn_ip_tcp** and **ncadg_ip_udp**. This environment variable should be used with caution. It will remove support for the protocol sequences which are not specified, and can cause communication failures when a client attempts to contact a server via a protocol sequence that it does not support.

Examples

```
export RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp
```

RPC_UNSUPPORTED_NETADDRS

Purpose

Prevents the RPC runtime from using the specified IP interfaces.

Synopsis

```
export RPC_UNSUPPORTED_NETADDRS=ipaddress:ipaddress
```

Description

This environment variable is to be used in situations where TCP/IP network interfaces are configured which you do not want DCE to use. It controls which of the local TCPIP interfaces will be used by the RPC runtime. The default is to use all configured TCPIP interfaces. The effect is that it controls how a server registers itself in the CDS database and endpoint map by masking out one or more networks through TCP/IP addresses.

This is useful in a machine that has multiple network adapters where the DCE traffic should be excluded from some of the networks. For example, consider a server machine that has one FDDI network connection for normal day-to-day traffic and is also

connected to two ethernet networks that are used only for X-station traffic. If a DCE server is started on this machine, it will register all three addresses in the CDS namespace and also in the dced endpoint map. This means that all machines on the FDDI network that want to communicate with this server have to have valid routing interfaces to the ethernet networks because when querying CDS for an address to the server, CDS could return one of the ethernet addresses to a machine that is only on the FDDI ring.

Suppose the machine described above has the following interfaces:

Interface	Address
en0	125.46.78.91
en1	125.46.125.91
fi0	9.25.47.91

The following example uses the **RPC_UNSUPPORTED_NETADDRS** environment variable to eliminate both ethernet networks by address:
`RPC_UNSUPPORTED_NETADDRS=125.46.78.91:125.46.125.91`

Examples

```
export RPC_UNSUPPORTED_NETADDRS=129.46.78.9
```

RPC_UNSUPPORTED_NETIFS

Purpose

Prevents the RPC runtime from using the specified IP interfaces.

Synopsis

```
export RPC_UNSUPPORTED_NETIFS=if_0:if_1
```

Description

This environment variable is to be used in situations where TCP/IP network interfaces are configured which you do not want DCE to use. It controls which of the local TCPIP interfaces will be used by the RPC runtime. The default is to use all configured TCPIP interfaces. The effect is that it controls how a server registers itself in the CDS database and endpoint map by masking out one or more networks through TCP/IP interfaces.

Examples

```
export RPC_UNSUPPORTED_NETIFS=en0:en1
```

SECURITY

Setting the DCE Security Environment variables is discussed in the following topics.

KRB5CCNAME

Purpose

Specifies the default credentials cache file.

Synopsis

`KRB5CCNAME=FILE:/var/dce/security/creds/dcecred_[`*XXXXXXXX*`]`

Description

This environment variable is set when you login to DCE, for example when using the dce_login command, or when using AIX/DCE security integration. KRB5CCNAME points to a file where

your DCE credentials obtained during login are cached. The *XXXXXXXX* portion of the file name is generated randomly each time you login.

Once you login and KRB5CCNAME is set, other programs you run can use these the cached DCE credentials (for as long as they are valid), without the need to re-authenticate to DCE to get your credentials. This is why KRB5CCNAME is said to refer to the default credentials cache.

By explicitly changing the value of KRB5CCNAME you can change your default DCE credentials (providing you have previously logged in to DCE and obtained another credentials file). Note, however, that changing the value of KRB5CCNAME does not change your DCE credentials for DFS -- these can only be changed by another login to DCE.

For more information on DCE credentials files, refer to the article on **dcecred_*** files in the *IBM DCE for AIX, Version 2.2: Administration Commands Reference* .

Examples

```
KRB5CCNAME=FILE:/var/dce/security/creds/dcecred_34210983
```

BIND_PE_SITE | TRY_PE_SITE

Purpose

Controls how a DCE client looks up the names of security replicas.

Synopsis

```
BIND_PE_SITE=[0]|[1]
TRY_PE_SITE=[0]|[1]
```

Description

When a DCE client needs to communicate with a security replica, it customarily looks up a replica name in the cds namespace. However, if the client contacts a replica frequently, the overhead of performing these cds lookups can be significant.

To improve performance, The **BIND_PE_SITE** and **TRY_PE_SITE** environment variables allow the client to lookup security replica names in the **/opt/dcelocal/etc/security/pe_site** file. The **pe_site** file contains the names and locations of the security replicas in the cell. Generally, locating a security replica using the **pe_site** file will be faster than looking in cds.

If neither **BIND_PE_SITE** or **TRY_PE_SITE** are set, or are set to `0`, then the client will locate a security replica using the traditional method of looking in the cds namespace.

If **TRY_PE_SITE** is set to `1`, the client will attempt to locate a security replica using the **pe_site** file. If no replica can be contacted, the client will next try to locate a replica by looking in the cds namespace.

If **BIND_PE_SITE** is set to `1`, the client will attempt to locate a security replica using only the **pe_site** file. If this fails, the client will not look in the cds namespace. Rather, the attempt to contact a security replica will fail.

If both **TRY_PE_SITE** and **BIND_PE_SITE** are set to `1`, the **TRY_PE_SITE** behavior takes precedence.

The **pe_site** file contains the names and locations of the security replicas in the cell. It is created when the DCE client is first configured into the cell. As security replicas become available and unavailable, the information on security replicas in the **pe_site** file may not be as current as in the cds namespace. For this reason, **BIND_PE_SITE** or **TRY_PE_SITE** should only be set when running programs which must contact a security replica frequently (for example, for frequent DCE login or registry operations).

The **pe_site** file is updated with current information on security replicas by the **dced** daemon, which updates it at regular intervals. It can also be updated by running the **chpesite** command.

Examples

```
BIND_PE_SITE=1
TRY_PE_SITE=0
```

Appendix C. 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.

The CDS Space

Figure 3 through Figure 5 on page 62 illustrate the CDS namespace of a DCE cell namespace. The subsections that follow provide a description of each entry.

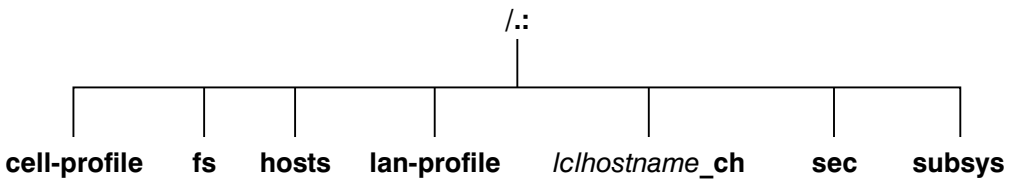

Figure 3. The Top-Level CDS Directory

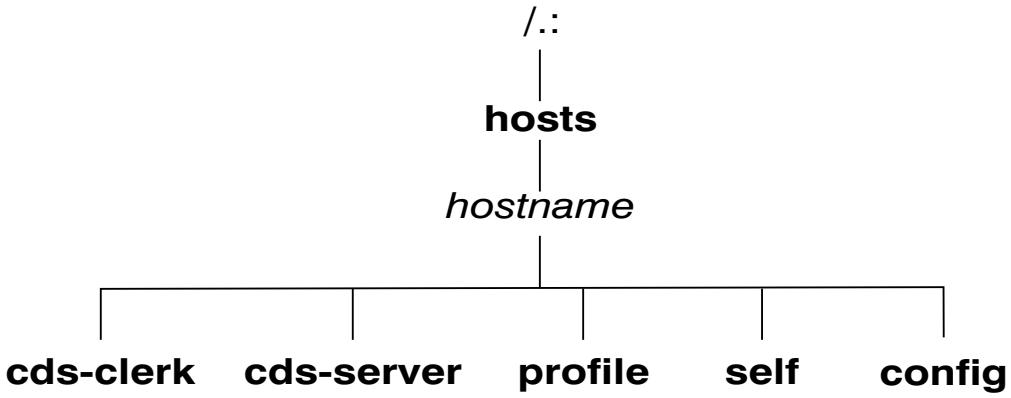

Figure 4. The CDS hosts Directory

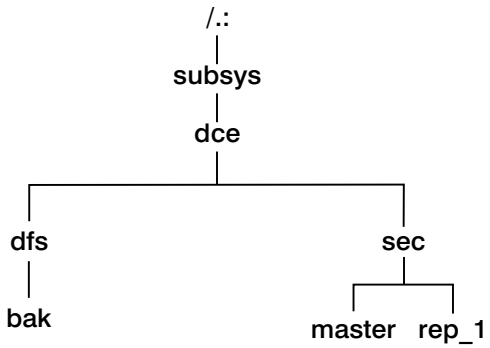

Figure 5. The CDS subsys Directory

The Top-Level CDS Directory

The following tables describe the namespace entries for ***/.:***, the top-level CDS directory.

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	

Name	**/.:/cell-profile**
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

Name	**/.:/lan-profile**
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

Name	**/.:/subsys**
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– – – }**

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– – – }**

Name	**/.:/hosts/***hostname*
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

Name	**/.:/hosts/***hostname***/profile**
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– }**

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 cril}**

Name	**/.:/hosts/***hostname***/config/hostdata**
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 cril}**
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 cril}**
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 cril}**
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 cril}**
Initial Object ACL	**{user hosts/***hostname***/self crwd}**

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

Name	**/.:/subsys/dce/dfs/bak**
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.

Name	**/.:/subsys/dce/sec/master**
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– }**

The Security Space

Figure 6 on page 72 through Figure 8 on page 73 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.

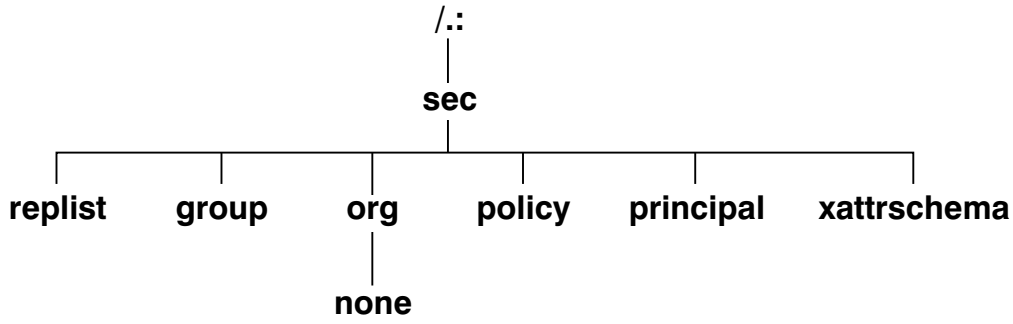

Figure 6. The Top-Level Security Directory

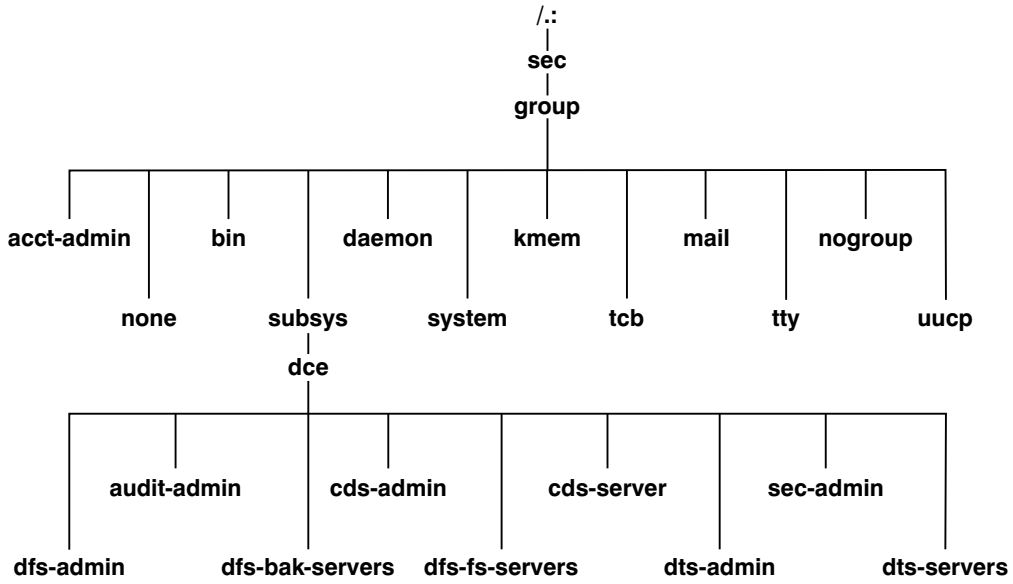

Figure 7. The sec/group Directory

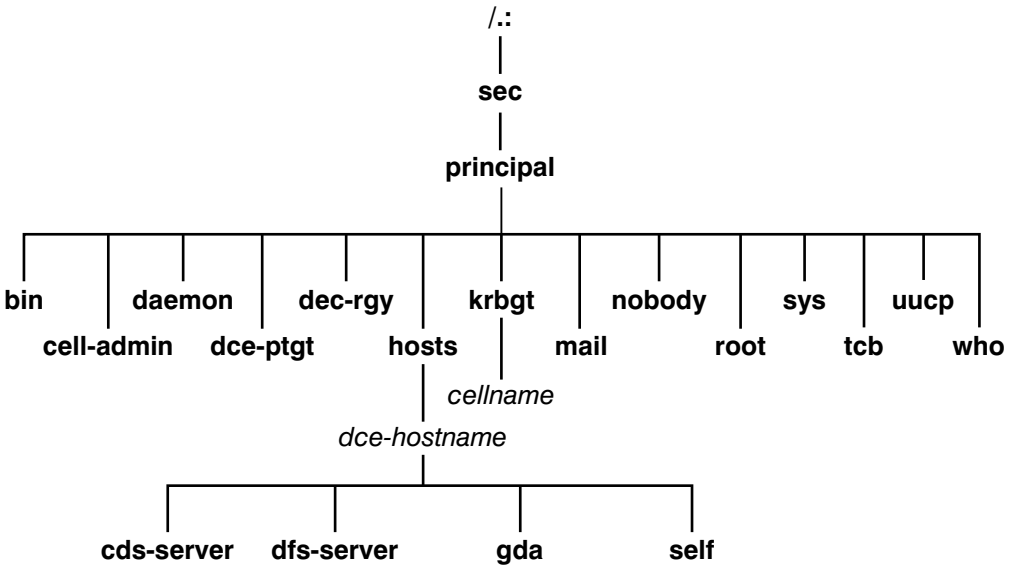

Figure 8. The sec/principal Directory

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.″

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– – – – – }**

Name	/.:/sec/principal
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– l} {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– – – – }**

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

Name	**/.:/sec/group/bin**
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	

Name	**/.:/sec/group/nogroup**
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

Name	/.:/sec/group/tcb
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

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

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

Name	**/.:/sec/group/subsys/dce/dfs-bak-servers**
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	

Name	/.:/sec/group/subsys/dce/dts-servers
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

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

Name	**/.:/sec/principal/mail**
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

Name	**/.:/sec/principal/tcb**
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

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***/gda**

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

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make them available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department LZKS
11400 Burnet Road
Austin, Texas 78758
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

- IBM
- AIX

DFS is a trademark of Transarc Corporation.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Index

Special Characters

A

B

C

D

T

U

V

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.