Administrator’s Guide

iPlanet Directory Access Router

Version 5.0

July 2001

Contents

List of Figures

List of Tables

About This Guide

The *Administrator's Guide* presents various deployment scenarios for iPlanet Directory Access Router (iDAR) and explains how to configure and maintain iDAR.

This preface has the following sections:

Audience for This Guide

The iDAR Administrator's Guide is written for administrators who will configure and operate one or more iDARs. This guide assumes that you have the following background:

- A general understanding of the Internet and LDAP.
- A general understanding of iPlanet Directory Server 5.x and its administration. You should be able to read and modify directory data.

What's in This Guide

The guide is organized into four parts:

- Part 1, "Introduction to iPlanet Directory Access Router"
- Part 2, "Console Based Administration"
- Part 3, "Command-Line Configuration"

- Part 4, “Appendixes”

Conventions Used In This Guide

This section explains the conventions used in this book.

`Monospaced font`—This typeface is used for any text that appears on the computer screen or text that you should type. It is also used for filenames, functions, and examples.

NOTE	Notes and Cautions mark important information. Make sure you read the information before continuing with a task.

The greater than symbol (>) is used as a separator for successive menu selections. For example, Object > New > User means that you should pull down the Object menu, drag the mouse down to highlight New, and drag the mouse across to the New submenu in which you must select User.

Throughout this book you will see path references of the form:

`<server-root>/idar-<hostname>/...`

where `<server-root>` is the default installation directory and `<hostname>` is the name of the host machine in which iDAR is installed. For example, if the installation directory is `/usr/iplanet/servers` and the hostname of the machine is `testmachine`, the actual path would be:

`/usr/iplanet/servers/idar-testmachine/. . .`

All paths specified in this manual are in UNIX format. If you are using a Windows NT-based directory server, you should assume the NT equivalent file paths whenever UNIX file paths are shown in this guide.

Related Information

In addition to this guide, the iDAR documentation set includes the following:

- **iPlanet Directory Access Router Release Notes.** The release notes contain important information available at the time of the release of iDAR. New features and enhancements, known problems, and other late-breaking issues are addressed here. Read this document before you begin using iDAR.

- **iPlanet Directory Access Router Installation Guide.** This guide contains procedures for installing iDAR plus requirements and tuning information.

The document set, including guides for deployment, installation, administration, and customization, for iPlanet Directory Server can be found at the following location: `http://docs.iplanet.com/docs/manuals/directory.html`

Other useful iPlanet information can be found at the following Internet locations:

- iPlanet release notes and other documentation:
  `http://docs.iplanet.com/docs/manuals/`
- iPlanet product status:
  `http://www.iplanet.com/support/technical_resources/`
- iPlanet Professional Services information:
  `http://www.iplanet.com/services/professional_services_3_3.html`
- iPlanet developer information:
  `http://developer.iplanet.com/`
- iPlanet learning solutions:
  `http://www.iplanet.com/learning/index.html`
- iPlanet product data sheets:
  `http://www.iplanet.com/products/index.html`
- iPlanet product technical support
  `http://www.iplanet.com/support`

Support

Thank you for choosing iPlanet Directory Access Router. Before requesting assistance, please check the FAQs and technical notes at `http://www.iplanet.com/support/online`. If you have any questions, please visit Customer Service at `http://www.iplanet.com/support`.

Part 1

Introduction to iPlanet Directory Access Router

Chapter 1, “Overview of iDAR”

Chapter 2, “iDAR Deployment Scenarios”

Chapter 1

Overview of iDAR

This chapter introduces you to iPlanet Directory Access Router (iDAR). The chapter consists of the following sections:

Introduction

iPlanet Directory Access Router (iDAR) is an essential component of any mission-critical directory service for e-commerce solutions. iDAR is an LDAP application layer protocol gateway that offers enhanced directory access control, schema compatibility, and high availability using application layer load balancing and failover.

Functionally, iDAR is an “LDAP access router” located between LDAP clients and LDAP directory servers. Requests from LDAP clients can be filtered and routed to LDAP directory servers based on rules defined in the iDAR configuration. Results from the directory server can be filtered and passed back to clients, again based on rules defined in the iDAR configuration. This process is totally transparent to the LDAP clients, which connect to iDAR just as they would to any LDAP directory server.

iDAR is a unique product that provides high availability, security, and client compatibility features for both extranet and intranet directory infrastructures, including:

- Automatic load balancing
- Transparent server failover and failback
- Automatic referral following

- Extranet / intranet access control groups
- Secure client and server authentication
- Dynamic query and response filtering
- Dynamic schema mapping
- Directory-based or file-based configuration
- Configurable logging

iDAR coexists with and complements new and existing LDAP directory infrastructures and integrates seamlessly with directory-enabled applications already deployed in enterprise extranets and intranets. It can be deployed to leverage the existing investment in a customer's directory infrastructure. iDAR will inter-operate with any LDAP compliant directory server. iDAR will work with any LDAP enabled and conformant directory, whether it's a native LDAP directory, an LDAP enabled X.500 directory, or an LDAP enabled relational database.

iDAR implements the LDAPv3 Internet specification and also supports the older and less functional LDAPv2 specification for compatibility with already deployed directory-enabled client applications that use LDAPv2. iDAR runs as a separate system server process on UNIX and Windows NT platforms. The server is multi-threaded and can handle thousands of LDAP client requests while applying access control rules and protocol filtering rules to each request.

iDAR can help organizations protect their private directory information from unauthorized access, while making it safe for these organizations to publish their public information. iDAR can be used to configure a fine-grained, access control policy on LDAP directories, such as controlling who can perform different types of operations on different parts of the Directory Information Tree (DIT). iDAR can also be configured to disallow certain kinds of operations typically performed by web trawlers and robots to collect information.

Unlike a web proxy server, iDAR operates in a reverse proxy mode. It does not forward connections to arbitrary servers on the Internet from clients inside the firewall. Neither does it cache search results. The predominant reason for this is the problem of applying access controls to the data. This is currently only done in the LDAP directory server where the access controls are maintained. iDAR has no knowledge of the directory server access controls.

iDAR Feature Set

The iDAR feature set provides distinct functions: high availability, load balancing, failover, firewall-like security, and client-server compatibility.

High Availability

iDAR is designed to support high availability directory deployments by providing both automatic load balancing and automatic failover and failback among a set of replicated LDAP directory servers. For extranet and intranet environments it is often necessary to ensure that mission-critical directory-enabled clients and applications have 24x7 access to directory data. iDAR maintains connection state information for all directory servers that it knows about, and is able to dynamically perform proportional load balancing of LDAP operations across a set of configured directory servers. Should one or more directory servers become unavailable, the load is proportionally redistributed among the remaining servers. When a directory server comes back on line, the load is proportionally reallocated dynamically.

For example, suppose directory server A is configured to receive 40 percent of the LDAP client load, server B 20 percent, server C 20 percent and server D 20 percent. If directory server B fails, iDAR will recognize that server A is configured to carry twice the load of servers C and D, and will redistribute the 20 percent load from server B such that server A now receives 50 percent, server C 25 percent and server D 25 percent. When directory server B is recovered, iDAR will automatically detect this and revert back to the original load percentages configured across all four servers.

Network layer IP load balancing devices don't have access to the LDAP protocol layer. However, iDAR integrates load balancing with access control, query filtering, and query routing, and can make intelligent application layer access control and LDAP routing decisions.

Load Balancing

Load balancing must be configured in iDAR using the load balancing property described in "ids-proxy-sch-LoadBalanceProperty Object Class," on page 226, or in Chapter 7, "Defining and Managing Property Objects." Each back-end directory server that iDAR can communicate with is configured to receive a percentage of total client load. iDAR then automatically distributes client queries to different

back-end servers to meet the load criteria defined in the configuration. If a server becomes unavailable, iDAR distributes the load percentage of that server proportionally among the available servers based on their load percentage. iDAR starts rejecting client queries if all back-end LDAP servers become unavailable.

Load balancing in iDAR is session-based. This means that the decision function that chooses a particular server to which a client's queries will be directed is applied once per client session; in particular, at the start of the client session. All subsequent client queries in that session are directed to the server that was chosen at the beginning of the session.

The number of back-end LDAP servers that iDAR can load balance depends on several factors, such as the size of the host running iDAR, the network bandwidth available, the query mix that iDAR receives, the length of client sessions, and iDAR's configuration. In general, iDAR can support fewer servers if most sessions are short lived and queries are computationally intensive. Computationally intensive queries are those that require the inspection of the entire message such as when the attribute renaming feature described in "Attribute Renaming Property," on page 114 is used.

iDAR uses a monitor process to make health checks on its backend servers. This feature is automatically enabled if load balancing is used. iDAR makes an anonymous search operation for the Root DSE every 10 seconds for each of its backend directory servers. If one of them becomes unavailable or unresponsive, iDAR removes it for the active load balanced server set. When the server becomes available again, it is reintroduced in the set.

Failover

iDAR detects when a server becomes unavailable either when a connection attempt is returned with a connection refused error or when it times out. Since both these cases occur at the initial stages of the session, and no operations have yet been processed for that session, iDAR fails over to another server, provided one is available transparently. In the connect attempt timeout case, the client can experience significant delay in getting a response. If a connection between iDAR and a back-end server is abruptly lost, iDAR returns LDAP_BUSY error for all outstanding operations to the affected client. Subsequently, iDAR fails over that client session to another directory server.

In order to avoid iDAR becoming the single point of failure for your directory deployment, we recommend you use at least two iDARs with an IP appliance in front of it.

Security

iDAR provides flexible external directory access control facilities that enhance the basic access control provided by a directory server. The access control mechanisms allow different users and communities of users to be associated with specific access groups to which administrator-defined security restrictions and query filters will be applied. The administrator can control access to entries based on LDAP authentication information, IP address, domain name, and other criteria. The server also supports secure authentication using Internet standard security mechanisms, including both challenge response “protected password” authentication and digital-certificate-based strong authentication and data encryption. For protected password authentication, only SASL mechanisms without connection protection are supported. A table of supported mechanisms is in “Supported SASL Mechanisms,” on page 184. The mechanisms must also be supported by the back-end directory servers. Chapter 6, “Creating and Managing Groups” explains how to enable authentication using the SASL mechanism.

For strong authentication and data confidentiality, support is available for both the Netscape alternate port 636 Secure Sockets Layer (SSL) mechanism and the Internet standard Transport Layer Security (TLS) mechanism. RSA, Digital Signature Standard, DES, and Diffie Hellman cryptographic algorithms are also supported.

iDAR does not support strong authentication using the SASL EXTERNAL mechanism as it will be construed as a man-in-the-middle attack by the SSL protocol layer.

A significant security feature that iDAR provides is the protection of the number of connections established between LDAP clients and the LDAP directory server. You can protect your LDAP directory server from connection attacks by configuring iDAR to monitor a number of specific metrics: the number of simultaneous client operations, the number of operations a client can request per connection, and the number of connections for a particular client group. It also has the ability to time out inactive clients.

You can configure iDAR with specific threshold limits not to be exceeded for the given metrics. iDAR will monitor these metrics and ensure that the thresholds are not exceeded. iDAR keeps several metrics, such as the number of connections open from a particular host, the number of operations performed on a particular session, etc., to limit possible trawling of the directory and denial of service attacks. A detailed description of the configuration of these parameters are in “Configuring System Settings,” on page 63.

iDAR also limits trawling by disallowing certain kind of generic filter such as (cn=A*) or (cn>A). More details on how to configure filtering of filters is in Chapter 6, “Creating and Managing Groups.”

iDAR allows an authenticated client to change its access control to the directory service. This allows authenticated clients to have greater access to the directory information even if they are outside the secure network.

iDAR provides data protection by supporting Secure Socket Layer (SSL) transport protocol. You can, for example, configure iDAR so that all clients that access your directory services from outside the protected network are required to establish a SSL session. Details on configuring SSL in iDAR is given in “Configuring Security,” on page 179.

These features can help prevent “denial of service” attacks and “flood attacks” that are so commonplace in the industry today. If iDAR detects that a threshold has been reached, it will then start refusing connections to the directory server and prevent the directory server from being attacked and overwhelmed.

Client-Server Compatibility

iDAR makes query routing decisions based on LDAP distinguished names (DNs) and group access rights, including identifying mobile users based on authentication credentials. iDAR automatically follows LDAP referrals that may be returned by a directory server, in support of highly distributed and scalable directory services. Automatic referral following is a significant advantage for large-scale directory deployments where you must physically distribute directory information among a set of directory servers, but have the distributed directory appear to users as one logical directory. iDAR supports this type of deployment scenario by providing the ability to logically unify otherwise distributed directory data in support of scalable distributed directory services.

iDAR supports any compliant LDAPv2 or LDAPv3 client application. Support is provided for schema rewriting to accommodate client applications with fixed schemas that do not always match the directory server’s schema. For example, the Microsoft Outlook™ email client has a fixed schema that expects the directory server to implement Microsoft-defined attributes that may not match an enterprise’s more general schema requirements. The schema rewriting capability allows the directory system administrator to implement a general purpose enterprise schema, and then map specific elements of that schema dynamically into the set of attribute types that are required by the less functional client application. iDAR is otherwise schema agnostic and accepts any attribute types and object classes defined by a large set of standard and *ad hoc* industry schema definitions, including RFC1274, X.520, X.521, LIPS, PKIX, inetOrgPerson, and DEN.

Chapter 2

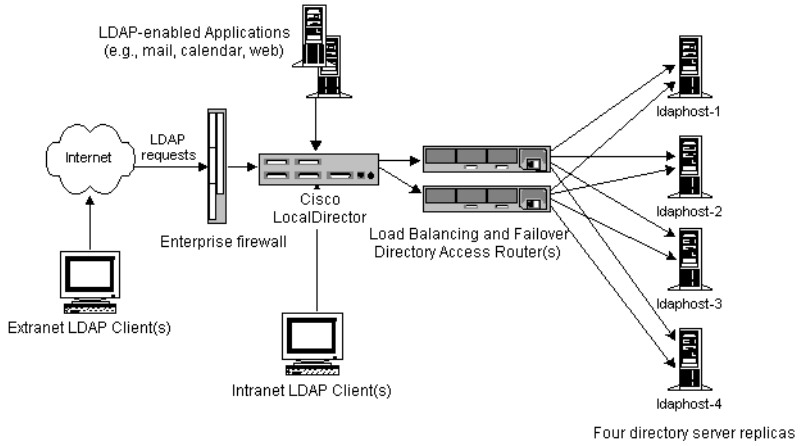
iDAR Deployment Scenarios

There are several ways you may want to deploy iPlanet Directory Access Router (iDAR), depending on your computing environment. This chapter describes and illustrates typical deployments, including:

An Internal High Availability Configuration

In the configuration shown in Figure 2-1, the customer has deployed an LDAP infrastructure for internal enterprise use only. There is no requirement for external network access to any of the enterprise LDAP services. The customer has deployed an enterprise firewall that will reject any access to internal LDAP services originating from outside the firewall. All client LDAP requests initiated internally must still go through iDAR via the Cisco LocalDirector (for high availability), which is shown here only as an example of IP packet switching that ensures clients have access to at least one iDAR. The customer prevents direct access to the directory servers by everyone except the hosts running iDARs; this can be achieved by using firewalls to protect the hosts running the directory server and iDAR.

Figure 2-1 Internal High Availability Configuration



A Distributed LDAP Directory Infrastructure

The following sections explain the role of iDAR in a distributed LDAP directory infrastructure:

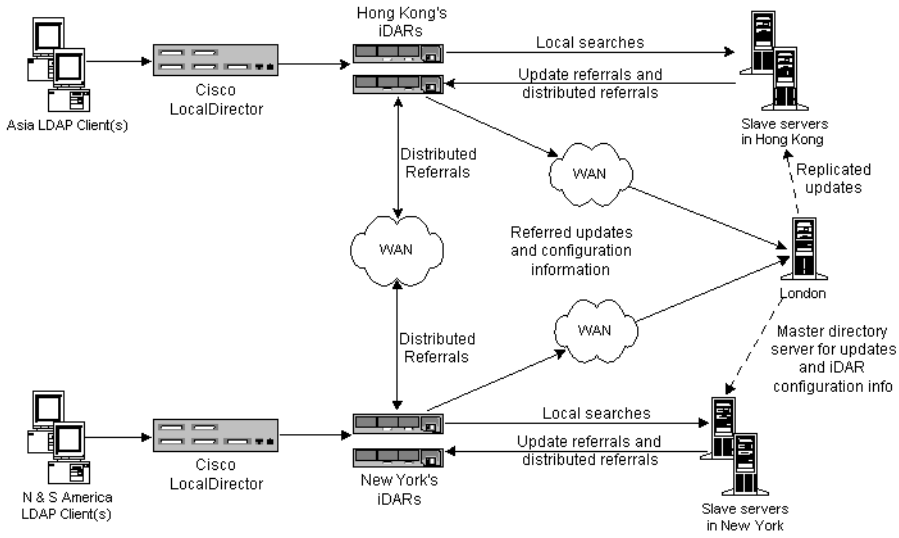
- Customer Scenario
- Customer Deployment
- LDAP Request Flow

Customer Scenario

In the configuration shown in Figure 2-2, a large financial institution is headquartered in London with data centers in London, New York, and Hong Kong. Currently, the vast majority of the data available to employees resides centrally in legacy RDBMS repositories in London. All access to this data from the financial institution's client community is via the Wide Area Network (WAN). The financial institution is experiencing scalability and performance problems with this centralized model and has decided to move to a distributed data model. The financial institution has also decided to deploy an LDAP directory infrastructure at the same time. The data in question is considered "mission critical" and should therefore be deployed in a highly available, fault tolerant infrastructure. An analysis of client application profiles has revealed that 95 percent of data accessed

by a geographical client community is specific to that community since the data is customer based. It is rare for a client in Asia to access data for a customer in North America but it does happen infrequently. The client community also has a need to update customer information from time to time.

Figure 2-2 A Distributed LDAP Directory Infrastructure



Customer Deployment

Given the profile of 95 percent local data access, the financial institution decided to distribute its LDAP directory infrastructure geographically. It deployed multiple directory consumer servers in each geographical location (i.e., Hong Kong, New York, and London; London consumer servers are not shown in the diagram). Each of these consumer servers is configured to hold the customer data specific to the location. Data for European and Middle East customers is held in the London consumer servers, data for North and South American customers is held in the New York consumer servers, and data for Asian and Pacific Rim customers is held in the Hong Kong consumer servers. With this deployment, the overwhelming data requirement of the local client community is located in the community. This provides significant performance improvements over the centralized model since client requests are processed locally, thereby reducing the network overhead; the

local directory servers are effectively partitioning the directory infrastructure, thereby providing increased directory server performance and scalability. Each set of consumer directory servers is configured to return referrals if a client submits an update request or if a client submits a search request for data located elsewhere.

LDAP Request Flow

Client LDAP requests are sent to the iPlanet Directory Access Routers via the Cisco LocalDirector. The LocalDirector product is shown here only as an example of IP packet switching that ensures clients always have access to at least one iDAR. The locally deployed iDAR's initially route all requests to the array of local directory servers holding the local customer data. The iDAR's are configured to load balance across the array of directory servers, thereby providing automatic failover and failback. Client search requests for local customer information are satisfied by a local directory and appropriate responses returned to the client via the iDAR. Client search requests for geographically "foreign" customer information are initially satisfied by the local directory server by returning a referral back to the iDAR. This referral contains an LDAP URL that points to the appropriate geographically distributed iDARs. The local iDAR processes the referral on behalf of the local client and sends the search request to the appropriate distributed iDAR. The distributed iDAR forwards the search request on to the distributed directory server and receives the appropriate response. This response is then returned to the local client via the distributed and the local iDAR's.

Update requests received by the local iDAR are also satisfied initially by a referral returned by the local directory server. Again, iDAR follows the referral on behalf of the local client but this time forwards the update request onto the supplier directory server located in London. The supplier directory server applies the update to the supplier database and sends a response back to the local client via the local iDAR. Subsequently, the supplier directory server will propagate the update down to the appropriate consumer directory servers.

All the iPlanet Directory Access Routers are configured to start up and look for their configuration in the supplier directory server. This allows you to distribute the iDARs geographically but manage their configurations centrally.

A Centralized LDAP Directory Infrastructure

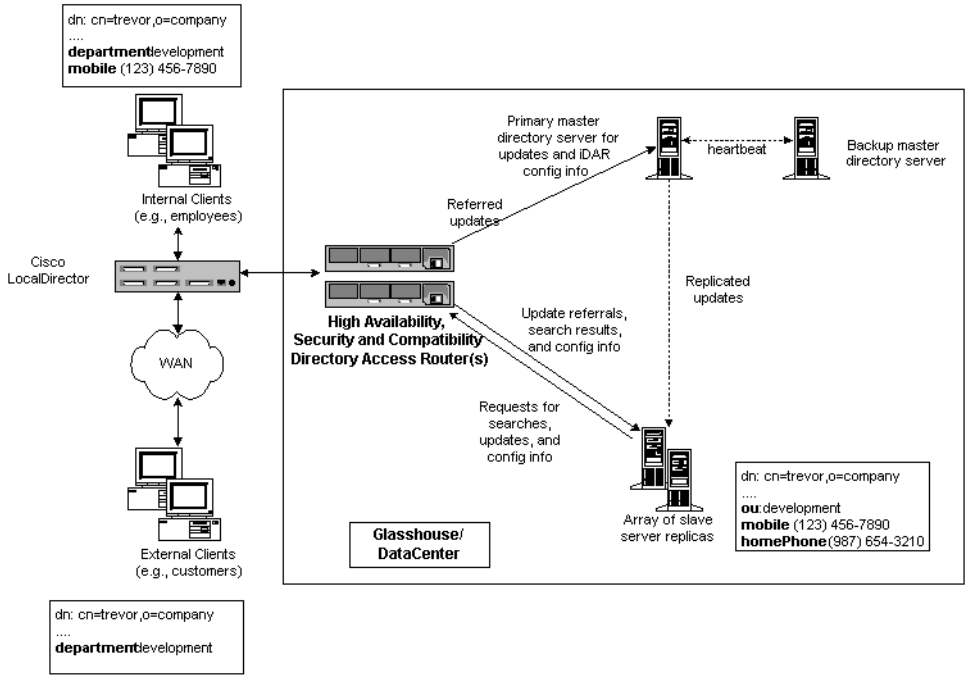
The following sections explain the role of iDAR in a centralized LDAP directory infrastructure:

- Customer Scenario
- Customer Deployment
- LDAP Request Flow

Customer Scenario

Figure 2-3 depicts a large global enterprise, with customers and employees distributed throughout the world, that wanted to deploy a corporate white and yellow pages (electronic phone book) to reduce the cost of printing a paper phone book, to increase the accuracy of the corporate information, and to reduce the use of environmental resources. The white and yellow pages information had to be available to both customers and employees with appropriate access controls. They also had to be available 24x7 and were classified as mission critical due to customers and employees being distributed throughout the world across all time zones.

Figure 2-3 A Centralized LDAP Directory Infrastructure



Customer Deployment

The global enterprise decided to deploy a centralized LDAP directory infrastructure to support the deployment of the white and yellow pages. A centralized deployment was chosen in this instance because the white and yellow pages are for corporate employee information only. This was not to be a customer database although the intent was for customers to have access to some of the information. It was decided that the projected size of the directory database (~200,000 entries) was not sufficient to require a more complex distributed deployment model since neither scalability nor performance were anticipated to be a problem.

Since there was a high availability requirement, the enterprise decided to deploy multiple consumer directory server replicas supplied by a single supplier directory server. To remove the single point of failure introduced by a single supplier directory server, the enterprise deployed a backup supplier directory server.

The iPlanet Directory Access Routers were deployed for three different reasons. First, to provide the load balancing and automatic failover and failback between all LDAP clients and the array of directory server replicas. Second, to be able to differentiate between external and internal clients and to set appropriate access controls accordingly. Third, to provide compatibility between the LDAP clients using the white and yellow pages and the directory servers themselves. In addition to utilizing a custom-built white and yellow pages application, the LDAP clients also used a number of off-the-shelf LDAP enabled applications that came with fixed schema requirements. These schema requirements did not always match the directory schema designed by the enterprise, therefore requiring some basic schema attribute mapping. In addition, not all the LDAP enabled applications used by the clients were capable of processing referrals received from the directory servers correctly. The iDARs were configured to follow these referrals on behalf of the clients.

LDAP Request Flow

All client requests, whether from internal or external clients and whether search requests or update requests, are sent to the iDAR's via the Cisco LocalDirector. The LocalDirector product is shown here only as an example of IP packet switching that ensures clients always have access to at least one iDAR. Multiple iDAR's are deployed to ensure there is no single point of failure. The iDAR's load balance all the requests received from the clients across all the consumer directory servers in the array. The iDAR's will also detect the failure of any of the consumer servers and failover to the available consumer servers in the array.

Since consumer servers are read-only replicas, they are configured to return an LDAP referral when an update request is received from a client. This referral contains an LDAP URL pointing to the supplier directory server. When the directory server returns the referral, the iDAR recognizes it and follows the referral on behalf of the client. It binds to the supplier directory server and sends the update request to it. The supplier directory server applies the update to the supplier database and sends a response back to the client via iDAR. Subsequently, the supplier directory server will propagate the update down to the appropriate consumer directory servers.

Search requests sent by clients get routed via the iDARs to the array of consumer directory server replicas. iDARs can be configured to "inspect" these search requests before sending them on to the directory servers and filter out any requests that don't meet the access control and security rules configured for a particular client group and perform any necessary mappings. iDAR can also be configured to "inspect" the search result returned by the directory server and again perform appropriate filtering and mapping. In the example shown in Figure 2-3, both

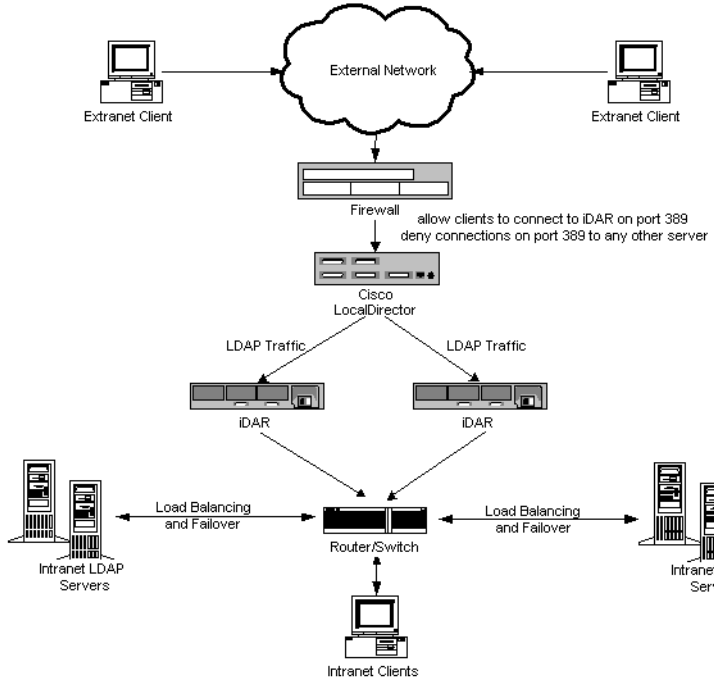
internal and external clients have requested a search for the entry belonging to "Trevor." These inbound requests are treated identically by iDAR irrespective of the client type. The directory server executes the request successfully and returns the entry for "Trevor" back to the iDAR. iDAR has been configured to manipulate the search result differently depending on whether the original request came from an internal or external client. In the case of the external client, both the mobile phone number and the home phone number fields in the entry are filtered out since they are deemed to be data inappropriate for customers. Note also that the ou: development attribute/value pair has been mapped to department: development. This is necessary because one of the applications the client is using to access the directory (e.g., Outlook, Outlook Express) has fixed schema elements that do not match the schema elements deployed in the enterprise directory servers. In the case of the internal client it was determined that the mobile phone number was an important data element to share among employees whereas the home phone number was not. So for internal clients, iDAR is configured to filter out only the home phone number and to permit the client to see the mobile phone number. Note the same mapping of the ou attribute to the department attribute is also performed.

All iPlanet Directory Access Routers are configured to start up and look for their configuration in the supplier directory server. This allows for the management of multiple iDAR configurations centrally from a directory.

Deploying iDAR with a Single Firewall

Your organization's firewall must be configured like that shown in Figure 2-4 to allow only LDAP clients to access the machine and port on which the iDAR is running. Typically, LDAP clients will connect on TCP port 389. This will protect the host running iDAR from clients who may try to gain unauthorized access to it. Also, placing the host running the proxy server on its own LAN, by using the router switch, will protect your internal network from denial of service attacks such as flooding your network with unnecessary traffic. The firewall should also disallow LDAP access to the machine(s) and port(s) on which the LDAP Directory Server(s) are "hiding," thereby protecting the LDAP directory database(s).

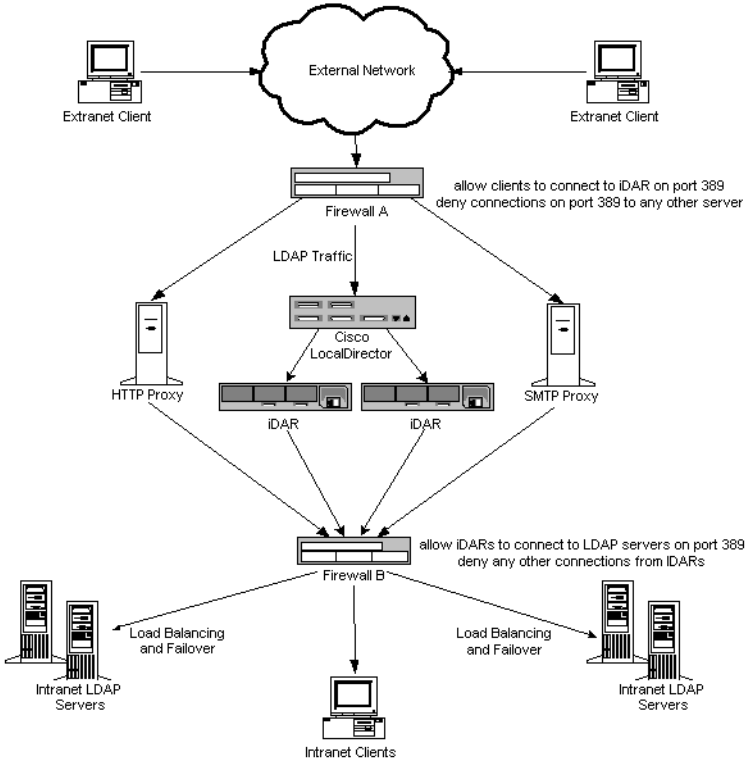
Figure 2-4 iDAR Setup with One Firewall



Deploying iDAR with Two Firewalls

The configuration shown in Figure 2-5 has all the benefits of the configuration shown in Figure 2-4 with some additional security. Installing two firewalls creates a zone of control around the “proxies,” allowing the site administrator to rate limit traffic from the external networks. It also ensures that a compromise of one of the “proxy” servers cannot be used directly to attack other machines in the interior network. Firewall A would be configured to allow only incoming packets if the destination IP address is that of the proxy handling that TCP or UDP protocol. Firewall B would be configured to only allow packets from the proxy machines that are appropriate to the servers that the proxy needs to access.

Figure 2-5 iDAR Setup with Two Firewalls



Part 2

Console Based Administration

Chapter 3

Introducing iDAR Consoles

After installing iPlanet Directory Access Router (iDAR), you first configure it to function with your directory deployment, and then on, closely monitor its activities. In administering iDAR, you perform server-specific tasks such as starting, stopping, and restarting the server; creating groups; setting up the server to identify certain events and execute appropriate actions; changing configuration; performing any routine server maintenance tasks; and monitoring logs.

To enable you to accomplish these server-specific tasks quickly and easily, iDAR provides GUI-based administration tools, called the *iDAR Server Console* and *iDAR Configuration Editor Console*, both of which are accessible from within the iPlanet Console. This chapter provides an overview of both iPlanet Console and iDAR consoles.

The chapter has the following sections:

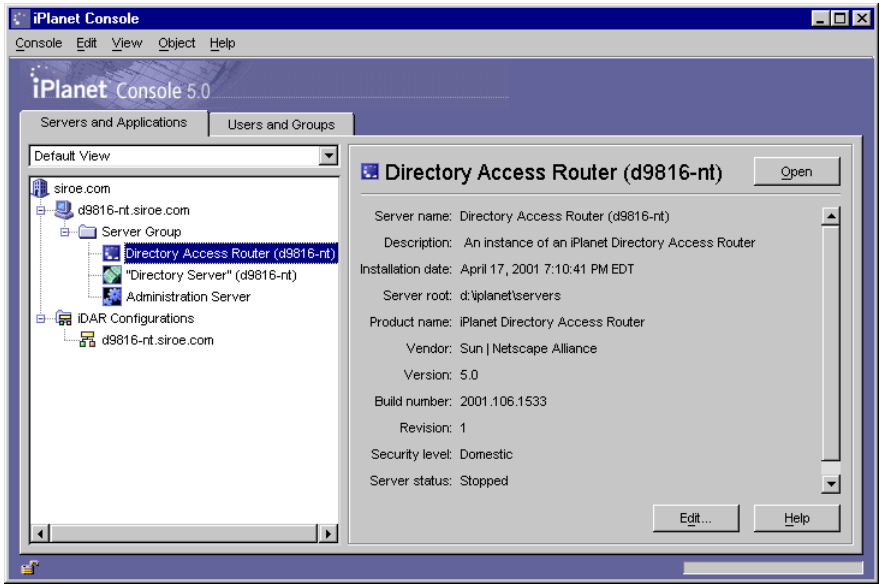
NOTE You can use iPlanet Console for managing various network resources. However, this chapter's focus is on using the iPlanet Console for iDAR administration only. For complete information about the iPlanet Console, see *Managing Servers with Planet Console*, which is included with the iDAR documentation. You can also get a copy of this book from this site: `http://docs.iplanet.com/docs/manuals/console.html`

Getting Started with iPlanet Console

The iPlanet Console is a stand-alone Java application that provides a GUI-based front end to all network resources registered in an organization's *configuration directory*. This unified administration interface simplifies network administration by supplying access points to all iPlanet version 5.x server instances installed across a network. Similarly, it simplifies basic user and group management by providing a unified administration interface to the user directory.

Figure 3-1 shows the "Servers and Applications" tab of the iPlanet Console with an iDAR instance selected.

Figure 3-1 iPlanet Console: Servers and Applications Tab



Servers and Applications Tab

For any given instance of the iPlanet Console, the limits of the network it can administer are defined by the set of resources whose configuration information is stored in the same configuration directory—that is, the maximum set of hosts and servers that can be monitored from the iPlanet Console. The *superadministrator* (the person who manages the configuration directory) can set access permissions on all

network resources registered in the configuration directory. Thus, for a given administrator using the iPlanet Console, the actual number of visible hosts and servers may be fewer, depending on the access permissions set by the superadministrator.

The “Servers and Applications” tab displays all servers registered in a particular configuration directory, giving you a consolidated view of all the server software and resources under your control. What you control is determined by the access permissions the superadministrator has set up for you.

From this view, you can perform tasks across arbitrary groups or a cluster of servers in a single operation. In other words, you can use the “Servers and Applications” tab to manage a single server or multiple servers that are installed on different ports on one machine. Also, you can access individual server consoles (or administration interfaces) by double-clicking the icons for the corresponding server instance entries (SIEs).

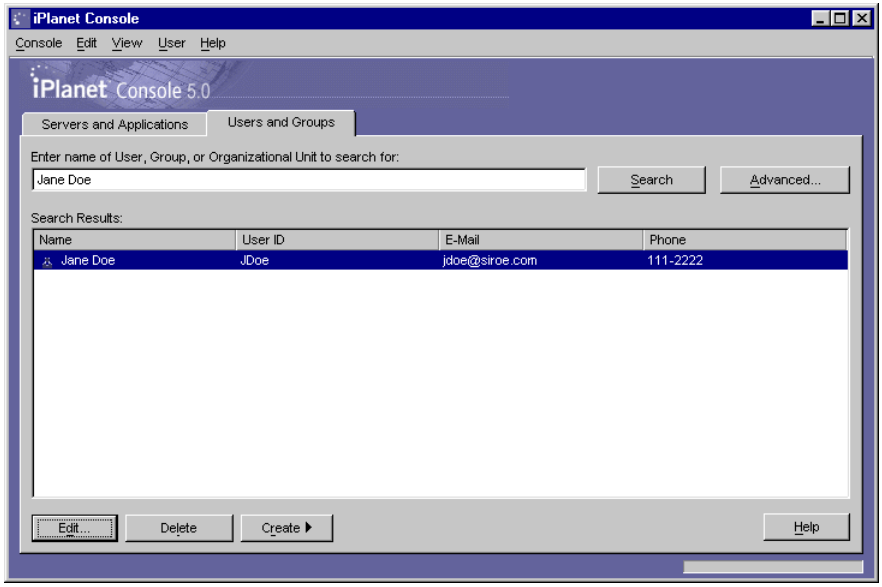
You can accomplish various iDAR-specific tasks from the “Servers and Applications” tab:

- Launch the iDAR Server Console.
- Launch the iDAR Configuration Editor Console (so that you can configure a group of iDARs).
- Set access permissions for iDAR.
- Launch the Administration Server Console (so that you can configure an Administration Server instance for administering iDAR).

Users and Groups Tab

The “Users and Groups” tab (shown in Figure 3-2) manages user accounts, group lists, and access control information for individual users and groups. All applications registered within the iPlanet Console framework share core user and group information in the user directory, which typically is a global directory for corporatewide user data.

Figure 3-2 iPlanet Console: Users and Groups Tab



From this tab, you can accomplish various user- and group-specific tasks, such as these:

- Add, modify, and delete user and group information in the user directory.
- Search for specific user and group entries in the user directory.

iPlanet Administration Server

iPlanet Administration Server is a web-based (HTTP) server that enables you to configure all your iPlanet servers, including iDAR, via the iPlanet Console. Administration Server (and the configuration directory) must be running before you can configure any of these servers. Administration Server is included with all the iPlanet servers and is installed when you install your first server in a *server group*. A server group refers to servers that are installed in a server root directory and that are managed by a single instance of Administration Server.

You access Administration Server by entering its URL in the iPlanet Console login screen; see “Step 1. Log In to the iPlanet Console,” on page 46. This URL is based on the computer hostname and the port number you chose when you installed iDAR. The format for the URL looks like this:
`http://<machine_name>.<your_domain>.<domain>:<port>`

Whenever you try to gain access to Administration Server, you will be prompted to authenticate yourself to the configuration directory by entering your user ID and password. These are the *administrator* user name and password that you specified when you installed iDAR (or the first server in the server group) and Administration Server on your computer. Once Administration Server is running, you can use the iPlanet Console to administer all servers in that group, including iDAR.

For complete details about Administration Server, see *Managing Servers with iPlanet Console*. To locate an online version of this book in your iDAR installation, open this file: `<server-root>/manual/en/admin/ag/contents.htm`

You can also get the latest version of this book from this site:

`http://docs.iplanet.com/docs/manuals/console.html`

Starting Administration Server

The iDAR installation program automatically starts the instance of Administration Server that you identified during installation for monitoring iDAR. If you stopped Administration Server after iDAR installation, you must start it before you can administer iDAR from the iDAR Console.

You can start Administration Server from the command line or from the Windows NT Services panel.

- To start Administration Server from the command line:

  At the prompt, enter the following line: `<server-root>/start-admin`

- Administration Server runs as a service in a Windows NT system. You can use the Windows NT Services panel to start the service directly.

All the above-mentioned methods start Administration Server at the port number you specified during installation. Once the server is running, you can use the iPlanet Console to access iDAR.

Stopping Administration Server

It is good security practice to shut down Administration Server when you are not using it. This minimizes the chances of someone else changing your configuration. You can shut down the server from the iPlanet Console, the command line, or the Windows NT Services panel.

- To shut down Administration Server from the iPlanet Console:

  a. Log in to the iPlanet Console (see “Step 1. Log In to the iPlanet Console” on page 46).

b. In the “Servers and Applications” tab, locate the Administration Server instance that you want to shut down, and double-click the corresponding entry.

The Administration Server Console appears.

c. In the Tasks tab, click Stop the Server.

- To shut down Administration Server from the command line:

  At the prompt, enter the following line: `<server-root>/stop-admin`

- Administration Server runs as a service in a Windows NT system; you can use the Windows NT Services panel to stop the service directly.

Accessing the iDAR Consoles

To perform any of the iDAR-administration tasks from the iDAR consoles, you need to open it first.

- Step 1. Log In to the iPlanet Console
- Step 2. Open the Appropriate iDAR Console

Step 1. Log In to the iPlanet Console

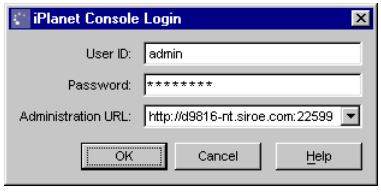
You can launch and use the iPlanet Console only when the corresponding configuration directory and Administration Server are running. If the servers are not running, go to the command line and start them. For information on starting Administration Server from the command line, see “Starting Administration Server,” on page 45. For information on starting the configuration directory, check the iPlanet Directory Server documentation.

When you launch the iPlanet Console, it displays a login window. You are required to authenticate to the configuration directory by entering your administrator’s ID, your password, and the URL (including the port number) of the Administration Server representing a server group to which you have access. You cannot use the iPlanet Console without having access privileges to at least one server group on your network.

1. Open the iPlanet Console application by using the appropriate option:

   - For local access on a UNIX machine, at the command-line prompt, enter the following line: `<server-root>/start-console`
   - For local access on a Windows NT machine, double-click the iPlanet Console icon on your desktop; this icon was created when you installed your first iPlanet server.

   The iPlanet Console Login window appears.



2. Authenticate yourself to the configuration directory.

   User ID. Type the *administrator ID* you specified when you installed Administration Server on your machine. You installed Administration Server either when you installed your first iPlanet server or as a part of iDAR installation.

   Password. Type the *administrator* password that you specified when you installed Administration Server on your computer during iDAR installation.

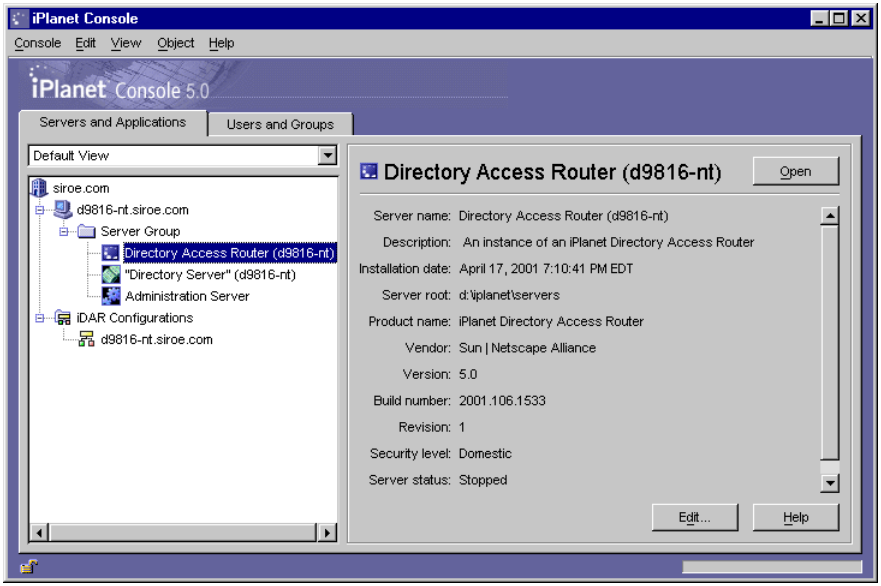
   Administration URL. This field should show the URL to Administration Server. If it doesn't or if it doesn't have the URL of Administration Server that you want, type the URL in this field. The URL is based on the computer host name and the Administration Server port number you chose when you installed iDAR. Use this format:

   `http://<machine_name>.<your_domain>.<domain>:<port_number>`

   For example, if your domain name is `siroe` and you installed Administration Server on a host machine called `myHost` and specified port number `12345`, the URL would look like this: `http://myHost.siroe.com:12345`

3. Click OK.

   The iPlanet Console appears with a list of all the servers and resources under your control.



Step 2. Open the Appropriate iDAR Console

In the iPlanet Console, you will notice that there are two entries for iDAR, one for the iDAR instance node and another for the iDAR Configurations node. The iDAR instance node corresponds to the iDAR server instance and the iDAR Configurations node corresponds to the configuration shared by multiple iDAR instances.

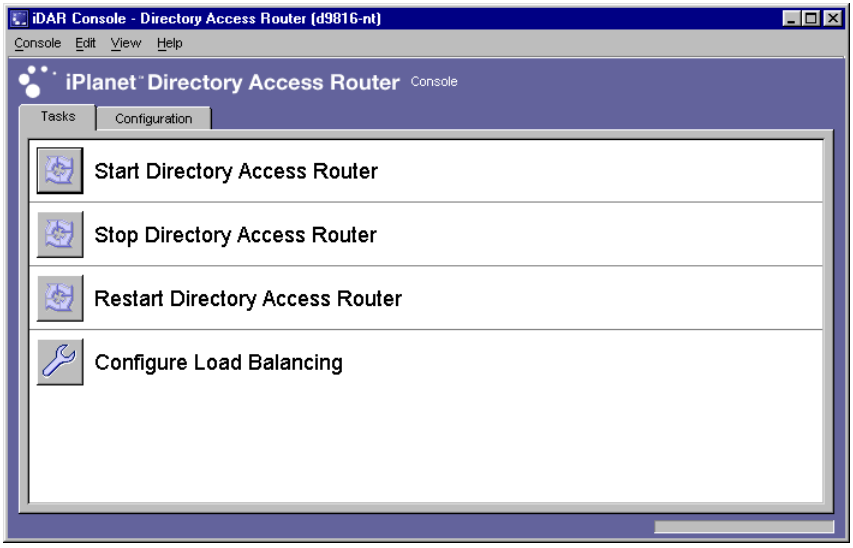
Each node is associated with a GUI-based administration interface:

- iDAR Server Console—This administration interface enables you to configure and manage an iDAR instance, for example to start it, to stop it, to specify configuration, to monitor logs, and so on. You can use the iDAR Server Console to access the server locally or remotely.
- iDAR Configuration Editor Console—The logic and system configurations can be shared by multiple iDAR instances. The ability of iDAR instances to share configuration information simplifies the task of managing a cluster of iDARs. The iDAR Configuration Editor Console is an administration interface that enables you to configure and manage a cluster of iDARs. Edits made via this interface affect all iDAR instances that use the edited configuration.

Opening the iDAR Server Console

Once you have logged in to the iPlanet Console, you can open the iDAR Server Console: in the navigation tree of the iPlanet Console, expand the hostname that contains the server group to which the iDAR instance belongs, expand the Server Group node, select the entry that corresponds to the iDAR instance of your interest, and click Open. The iDAR Server Console opens (Figure 3-3).

Figure 3-3 iDAR Server Console: Tasks Tab

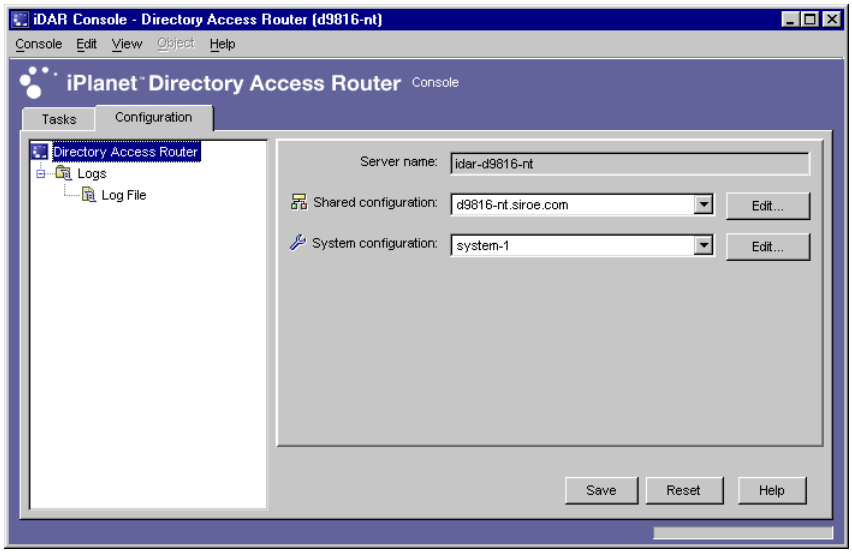


The iDAR Server Console to has two tabs—Tasks and Configuration—each addressing specific administrative areas.

The Tasks tab enables you to perform common tasks such as starting, stopping, and restarting the server, and distributing or balancing load among various LDAP directories. For details about starting, stopping, and restarting iDAR, see Chapter 4, “Starting, Restarting, and Stopping iDAR.” For details about load balancing, see Chapter 7, “Defining and Managing Property Objects.”

The Configuration tab (Figure 3-4) enables you to view and modify the configuration.

Figure 3-4 iDAR Server Console: Configuration Tab



The view displays settings related to how this specific instance of iDAR is configured.

Server name. A descriptive name for this instance of iDAR.

Shared configuration. Displays the currently selected configuration from which iDAR will draw its set of network groups, events, actions, properties, and system configurations. A new shared configuration can be specified by selecting an alternate entry from the list of the current set of shared configurations.

Edit. Displays the iDAR Configuration Editor associated with the configuration currently selected in the list box.

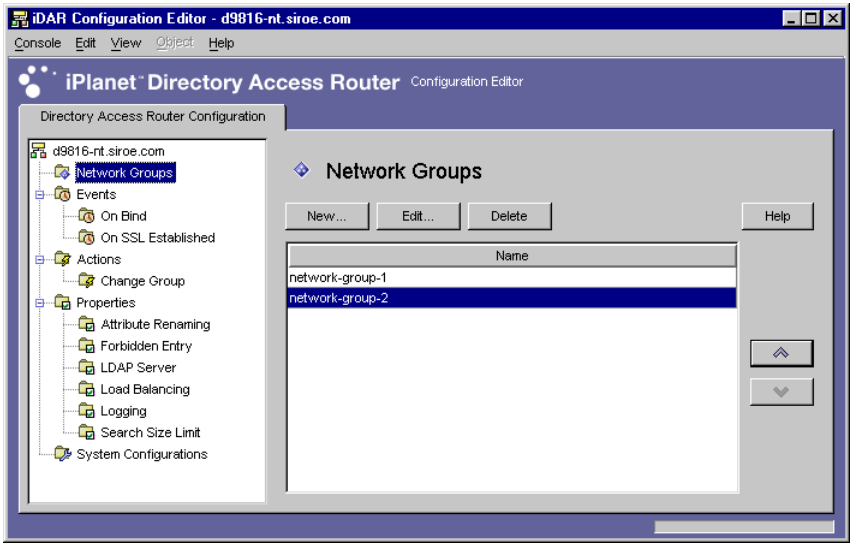
System configuration. Displays the available system configuration objects contained in the configuration displayed in “Shared configuration.” Changing the shared configuration replaces the items displayed in this list box. Additionally, the tree on the left side reflects information specified by the currently selected system property. Changing this setting may affect the makeup of the tree.

Edit. Displays the dialog for editing the System Configuration currently displayed in the list box.

Opening the iDAR Configuration Editor Console

Once you have logged in to the iPlanet Console, you can open the iDAR Configuration Editor Console: in the navigation tree of the iPlanet Console, expand the iDAR Configurations node, select the entry, and click Open. The iDAR Configuration Editor Console opens (Figure 3-5).

Figure 3-5 iDAR Configuration Editor Console



The navigation tree on the left side contains nodes for each of iDAR's basic configuration objects. Expanding one of the main nodes shows tree nodes for each of object subtype. Clicking a tree node displays a table on the right side containing all current objects of the type indicated by the selected tree node. Object tables whose ordering is important, for example, Network Groups, have a set of up and down buttons that allow individual objects to be raised or lowered in precedence.

Table 3-1 lists the configuration object types shown in the navigation tree.

Table 3-1 Configuration Objects in the iDAR Configuration Editor Console

Configuration Object Type	Description
Network Groups	Each Network Group object identifies a specific client community, and specifies the restrictions to enforce on clients that match that group. For details, see Chapter 6, "Creating and Managing Groups."

Table 3-1 Configuration Objects in the iDAR Configuration Editor Console *(Continued)*

Configuration Object Type	Description
Events	Event objects are used to specify conditions that occur at predetermined states. Conditions can be attached to certain events, on which, if satisfied, iDAR can take certain actions. For details, see Chapter 8, "Creating and Managing Event Objects."
Actions	Actions are used to specify actions to take when an event occurs. For details, see Chapter 9, "Creating and Managing Action Objects."
Properties	Properties are used to describe more specialized restrictions on the client. Each group object may include a set of properties defined by property objects. For details, see Chapter 7, "Defining and Managing Property Objects."
System Configurations	System Configurations are used to define instance-specific configurations for iDAR. For details, see Chapter 5, "Configuring System Parameters."

Chapter 4

Starting, Restarting, and Stopping iDAR

This chapter describes how to start, stop, and restart iPlanet Directory Access Router (iDAR) and how to check its current status.

The chapter has the following sections:

NOTE You can use the iDAR consoles only when the appropriate Directory Server (identified as the *configuration* directory) and Administration Server are running. Be sure to start Administration Server at the port you specified during iDAR installation. To minimize security risks, shut down Administration Server when you have finished using the iPlanet Console. For instructions on starting and shutting down Administration Server, see "iPlanet Administration Server," on page 44.

Starting and Stopping iDAR

Once iDAR is installed, it runs constantly, listening for and accepting requests; it runs as a UNIX daemon process or a Windows NT service normally started during system boot time.

You can start and stop iDAR in several ways:

- From the iPlanet Console (locally and remotely)

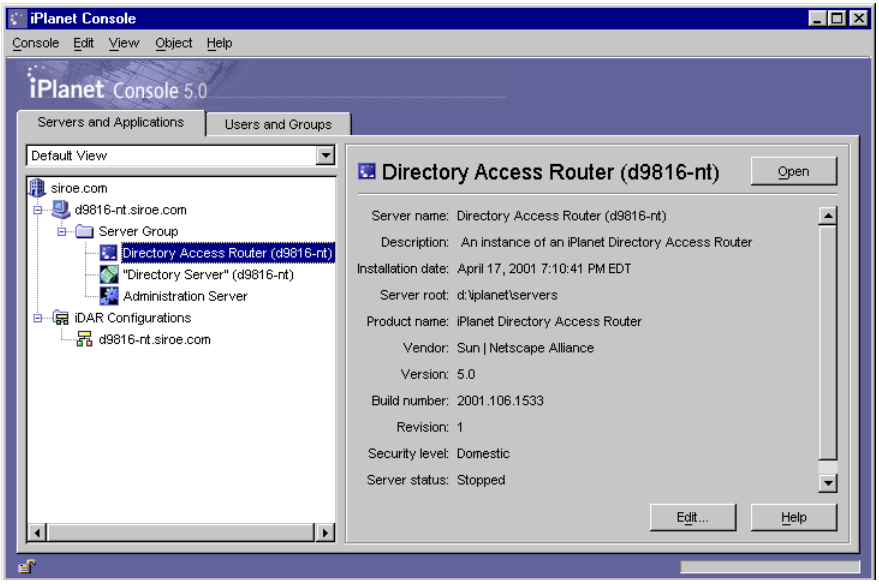
- From the command line (locally only)
- On a Windows NT system, from the Windows NT Services panel

Note that stopping iDAR shuts down all its components completely, interrupting service until the server is started again. If the host machine crashes or is taken off line, the server stops, and any requests it was servicing are lost. You need to start the server again to restore the service.

Starting and Stopping iDAR From iPlanet Console

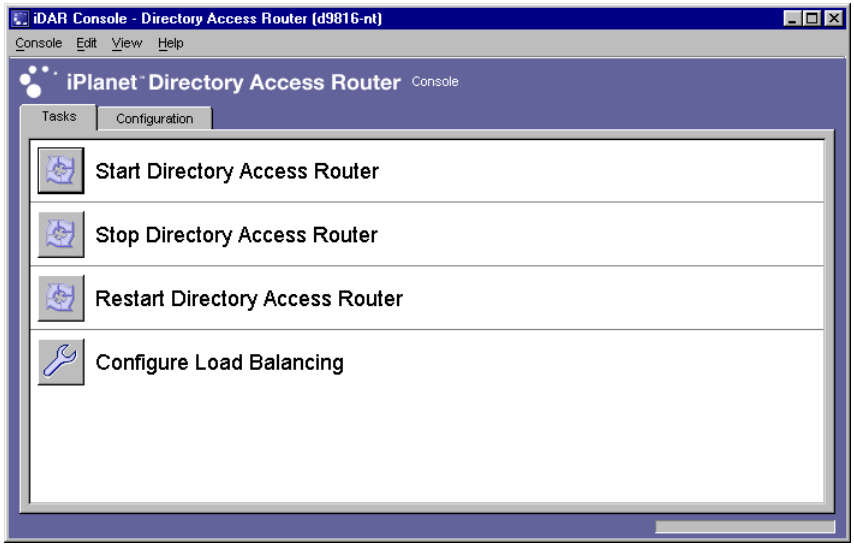
You can use the iPlanet Console to start and stop iDAR installed on a local or remote host. To start or stop iDAR:

1. Log in to the iPlanet Console (see “Step 1. Log In to the iPlanet Console,” on page 46).
2. In the “Servers and Applications” tab, expand the hostname and then the Server Group that contains the iDAR instance you want to start.
3. In the navigation tree, locate the iDAR instance you want to start or stop, select the corresponding entry, and click Open.



The iDAR Server Console opens.

4. In the Tasks tab, click Start Directory Access Router to start the server or Stop Directory Access Router to stop the server.



Starting and Stopping iDAR From Command Line

To start or stop iDAR from the command line:

1. Open a terminal window to your server.
2. In a UNIX system, log in as `root` if the server runs on ports less than 1024; otherwise, log in either as `root` or with the server's user account. (By default, if iDAR is run by `root`, it changes its user ID to nobody.)
3. At the command-line prompt, enter either of the following lines:

   To start iDAR: `<server-root>/idar-<hostname>/start-idar[.exe]`

   To stop iDAR: `<server-root>/idar-<hostname>/stop-idar[.exe]`

   `<server-root>` is the directory where the iDAR binaries are kept. You first specified this directory during installation.

   `<hostname>` is the name of the host on which this instance of iDAR is installed.

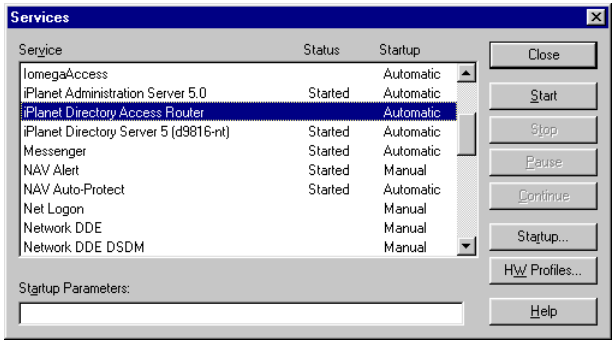
`.exe` specifies the file extension; this is required only when running the utility on a Windows NT system.

NOTE If iDAR is already running, the start-up command fails. Stop the server first using the `stop-idar` command, then use the `start-idar` command.

Starting and Stopping iDAR From Windows NT Services Panel

If you have installed iDAR on a Windows NT system, you can start and stop the server (as a service) from the Windows NT Services panel (see Figure 4-1). The iDAR service has the following name: `iPlanet Directory Access Router`.

Figure 4-1 iDAR Service Highlighted in the Windows NT Services Panel



To start or stop iDAR from the Windows NT Services panel:

1. On your desktop, select Start > Settings > Control Panel.
2. In the Control Panel window that appears, double-click Services.
3. Scroll through the list of services, and locate the service that corresponds to the iDAR instance.
4. To start the service, select the iDAR instance, and click Start. To stop the service, select the iDAR instance, and click Stop.

Restarting iDAR

Whenever you change the iDAR configuration, you must save your changes for them to be stored in the configuration directory. All configuration changes require that you restart iDAR after you save the changes. If restarting is required, the console prompts you accordingly.

During restart, iDAR re-reads its configuration and uses the new configuration for future connections. Client connections that are already established continue to use the old configuration until the clients disconnect. The restart function is only available on UNIX platforms. On Windows NT, restarting iDAR is equivalent to stopping and starting iDAR.

Note that changes to certain configuration parameters require that you stop iDAR and start it again; for instructions, see “Starting and Stopping iDAR,” on page 53. Table 4-1 lists these configuration parameters.

Table 4-1 Configuration Changes That Require iDAR Startup

UI Elements	Attributes
The Host field in the Settings tab of the System Configuration window; see “Configuring System Settings,” on page 63.	ids-proxy-con-listen-host
The Port field in the Settings tab of the System Configuration window; see “Configuring System Settings,” on page 63.	ids-proxy-con-listen-port
The “SSL port” field in the Settings tab of the System Configuration window; see “Configuring System Settings,” on page 63.	ids-proxy-con-ldaps-port
The “CA signed certificate file” field in the Encryption tab of System Configuration; see “Configuring System Settings,” on page 63.	ids-proxy-con-ssl-cert
The “Private key file” field in the Encryption tab of System Configuration; see “Configuring System Settings,” on page 63.	ids-proxy-con-ssl-key
The “Connection backlog” field in the Connections tab of the System Configuration window; see “Configuring System Settings,” on page 63.	ids-proxy-con-listen-backlog

You can restart iDAR in two ways:

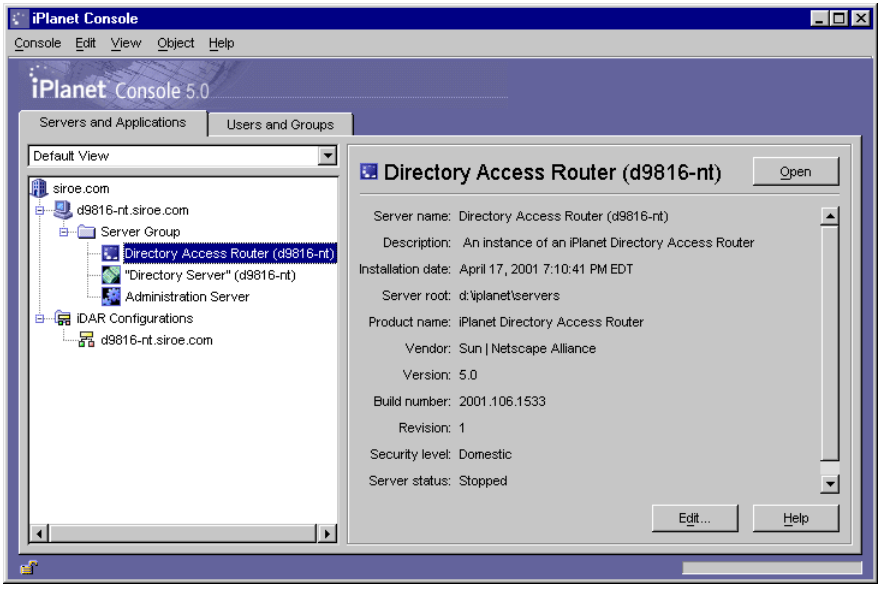
- From the iDAR Server Console (locally and remotely)
- From the command line (locally only)

Restarting iDAR From iPlanet Console

You can use the iDAR Server Console to restart an instance of iDAR installed on a local or remote host.

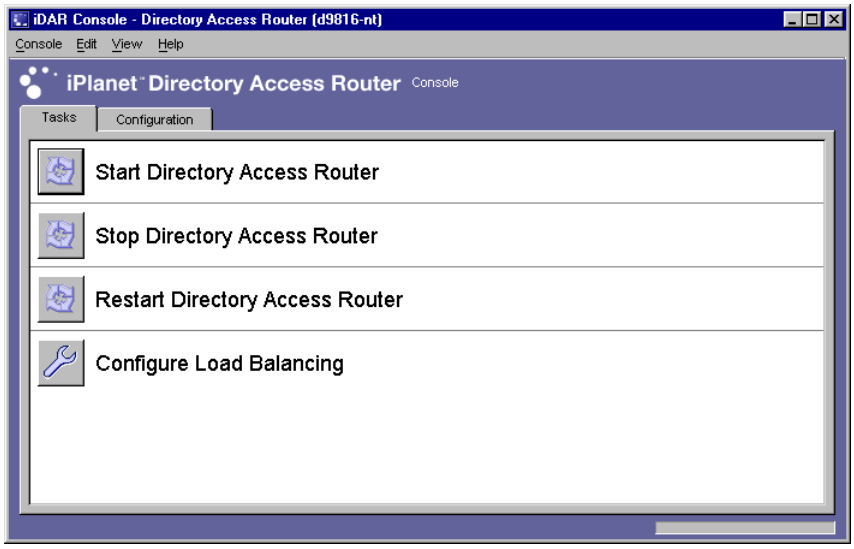
To restart iDAR from the iDAR Server Console:

1. If you're not already viewing the iDAR Server Console, log in to the iPlanet Console (see "Step 1. Log In to the iPlanet Console," on page 46).
2. In the "Servers and Applications" tab, expand the hostname and then the Server Group that contains the iDAR instance you want to restart.
3. In the navigation tree, locate the iDAR instance you want to start or stop, select the corresponding entry, and click Open.



The iDAR Server Console opens.

4. In the Tasks tab, click Restart Directory Access Router to restart the server.



Restarting iDAR From Command Line

To restart iDAR from the command line:

1. Open a terminal window to your server.
2. In a UNIX system, log in either as `root` or using the server's user account (if that is how you started the server).
3. At the command-line prompt, enter the following line:

   ```
   <server-root>/idar-<hostname>/restart-idar[.exe]
   ```

Checking iDAR System Status

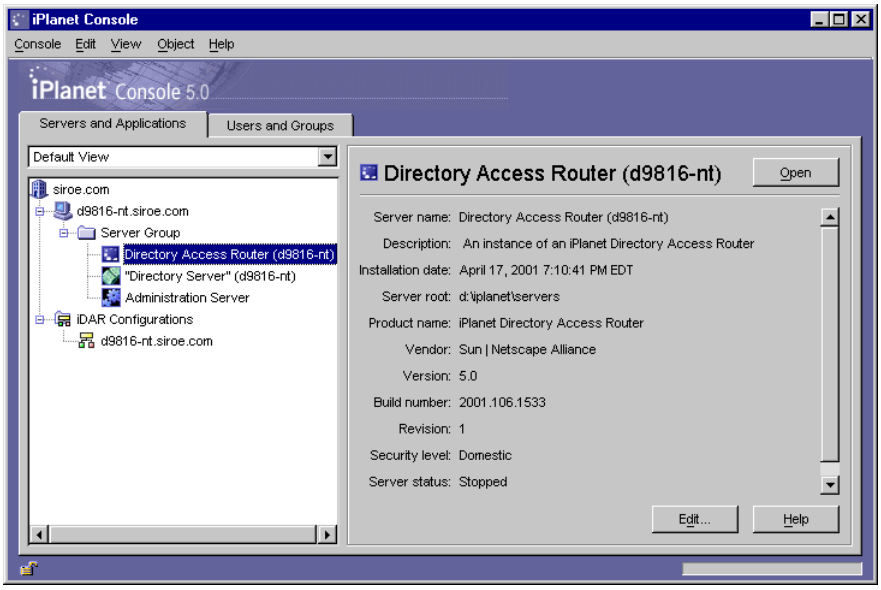
You can check whether a particular instance of iDAR is started or stopped in two ways:

- From the iPlanet Console (locally and remotely)
- From the command line (locally only)

Checking iDAR Status From iPlanet Console

You can use the iPlanet Console to find out whether a particular iDAR instance is running.

1. Log in to the iPlanet Console (see “Step 1. Log In to the iPlanet Console,” on page 46).
2. In the Servers and Applications tab, select the entry that corresponds to the iDAR instance you want to check.



3. In the right pane, check the Server Status field.

   If the selected instance of iDAR is running, the status will be *Started*. Otherwise it will be *Stopped* or *Unknown*.

Checking iDAR Status From Command Line

To find out whether a particular iDAR instance is running from the command line:

1. Open a terminal window to your server.
2. In a UNIX system, log in either as `root` or using the server’s user account (if that is how you started the server).

3. At the command-line prompt, enter the following line:

   `<server-root>/idar-<hostname>/status-idar[.exe]`

Chapter 5

Configuring System Parameters

System parameters are those that effect the functional behavior of iPlanet Directory Access Router (iDAR). This chapter explains how to specify system configuration and configure iDAR for TLS/SSL-enabled communication with LDAP clients and servers.

The chapter contains the following sections:

Configuring System Settings

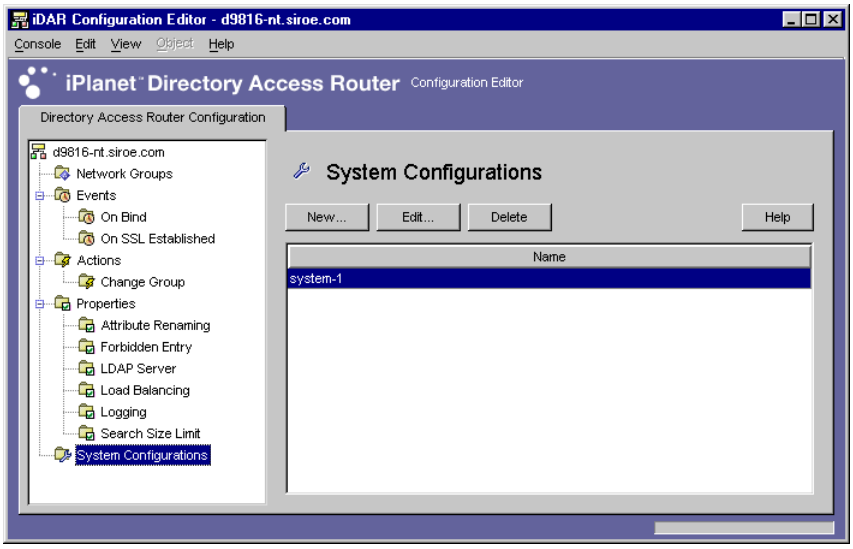
This section explains how to configure system-specific parameters of an iDAR instance.

Creating System Configuration Objects

To create an object for system configuration:

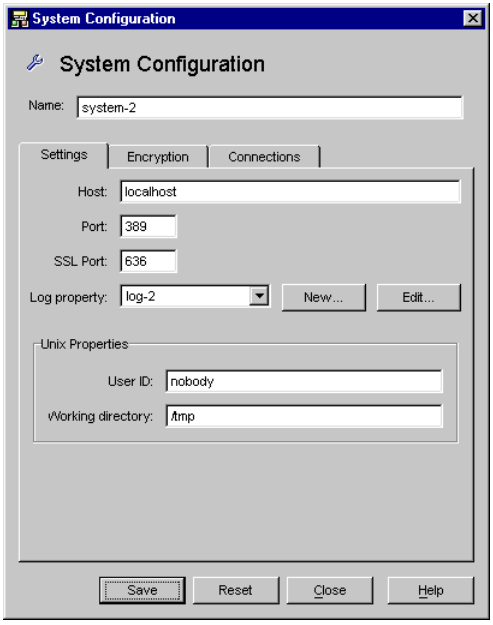
1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, select System Configurations.

   The right pane shows existing objects for the system configuration.



3. Click New.

   The System Configuration window appears.



4. In the Name field, type a name for the system configuration. The name must be a unique alphanumeric string.

5. In the Settings tab, specify general setting for this system configuration:

   Host. Enter the name of the host interface on which iDAR will listen for connections. This attribute is needed only if there are multiple network interfaces on the host running iDAR. By default, the hostname is set to "localhost," meaning iDAR will listen on all available network interfaces. Specifying "localhost" will permit shared system properties.

   Port. Enter the port number on which iDAR will listen for incoming connections. Legal values for this field are 1 through 65535. By default, the value is set to 389, as specified for LDAP. This port number must be different from that used by any other LDAP server running on the same host. On UNIX platforms the server must be started as root to listen on a port number below 1024.

   SSL port. Enter a value representing the port number on which to listen for LDAPS (LDAP over SSL) connections. By default, iDAR does not listen for connections from LDAPS clients. This value must be present to enable LDAPS connections from clients using this nonstandard function, with a value such as 636. This value must be different from the Host value. This option also requires TLS/SSL configuration, found on the Encryption tab.

   Log property. Choose a log property from which this system configuration will derive its custom logging information.

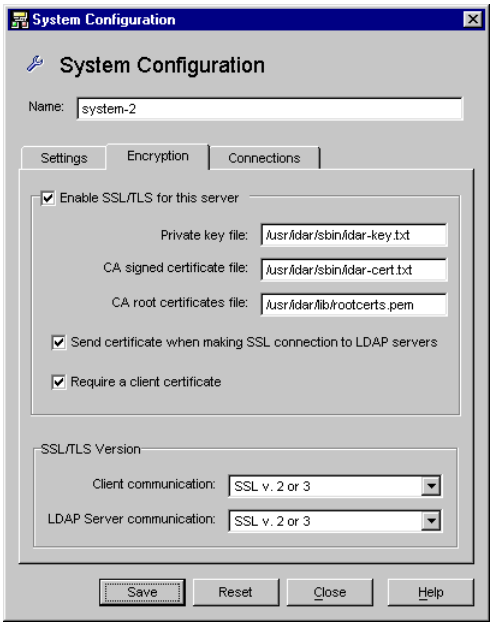
   Edit. Displays a dialog for editing the property currently displayed in Log Property.

   UNIX properties. This panel contains attributes that pertain to iDAR servers in a UNIX environment only.

   User ID. Enter the user ID under which iDAR will run. If iDAR was run as *root* then it will change its uid to the one specified here. The default is to switch to *nobody*. This option is not applicable on Windows NT.

   Working directory. Enter the directory from which iDAR should run. iDAR will change its working directory to the directory specified as value for this attribute on startup. The default is `/tmp`. This attribute only takes effect on platforms other than Windows NT.

6. Select the Encryption tab and configure iDAR for SSL-enabled communication.



The description of the parameters is as follows:

Enable SSL/TLS for this server. Select this box to enable SSL / TLS information needed by iDAR to listen over a secure connection. If an SSL port is specified, you must enable this setting and provide both the private key file and signed CA certificate file in order to save this configuration.

Private key file. Enter the path of the file on disk containing the private key associated with the server's certificate.

CA signed certificate file. Enter the path of the file on disk containing the server's own certificate signed by the CA. The certificate in the file must be in PEM format.

CA root certificate file. Enter the path of the file on disk containing trusted root certificates. These certificates must be in the PEM format.

Send certificate when making SSL connections to LDAP severs. Enable this setting if you want iDAR to send its certificate to the backend LDAP directory server when making a TLS connection. By default this setting is disabled.

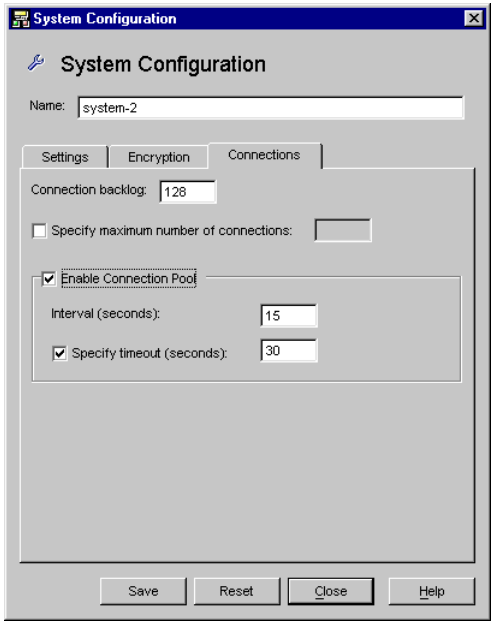
Require a client certificate. Enable this setting to specify that iDAR will require all clients that establish an SSL session to submit a certificate chain. iDAR will close the connection if a certificate chain is not submitted. Note that this option does not effect SSL sessions between iDAR and the backend servers. By default this setting is disabled.

SSL/TLS version. Select the SSL / TLS versions used for client and LDAP server communications with iDAR.

Client communication. Specify the SSL / TLS version iDAR will use between itself and the clients. The default is SSL version 2 or 3.

LDAP server communication. Specify the SSL / TLS version iDAR will use for connections between itself and backend LDAP servers. The default is SSL version 2 or 3.

7. Select the Connections tab and specify how iDAR should maintain its connections.



The description of the parameters is as follows:

Connection backlog. Enter a value greater than zero specifying the maximum number of outstanding connections in the listening socket's queue. The default is 128 connections. The maximum value depends on the underlying operating system configuration.

Specify maximum number of connections. Select the option and enter a value (greater than zero) specifying the maximum number of simultaneous client connections that iDAR will accept. To allow an unlimited number of simultaneous connections, do not select this option.

Enable Connection Pool. Enables the connection pool module with which iDAR will preconnect to the directory servers. The default for the setting is disabled. If the connection pool is enabled, iDAR will try to reuse existing connections to the backend LDAP servers. Switching on this option can give significant performance gain if the backend server is on a Wide Area Network (WAN).

Interval. Enter the number of seconds (greater or equal to one) specifying the interval in seconds at which iDAR will sample the incoming requests to anticipate future activity. The default is 15.

Specify timeout. Select the option and enter the number of seconds (greater or equal to zero) specifying the period of time in seconds after which an idle connection to an LDAP server will be terminated. If the checkbox is unchecked, no timeout will be applied. The default is 30. This value should be less than the idle connection timeout value of the backend LDAP server.

8. Click Save to save the object.

   The iDAR configuration is modified, and you are prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

9. Repeat Step 3 through Step 8 to create any additional objects.

10. Restart the servers; see "Restarting iDAR," on page 57.

NOTE Changes to the following will require stopping and starting iDAR: Host, Port, and "SSL port" fields in the Settings tab; the "CA signed certificate file" and "Private key file" fields in the Encryption tab; and the "Connection backlog" field in the Connections tab.

For instructions to stop and start iDAR, see "Starting and Stopping iDAR," on page 53.

Configuring iDAR for TLS/SSL-Enabled Communication

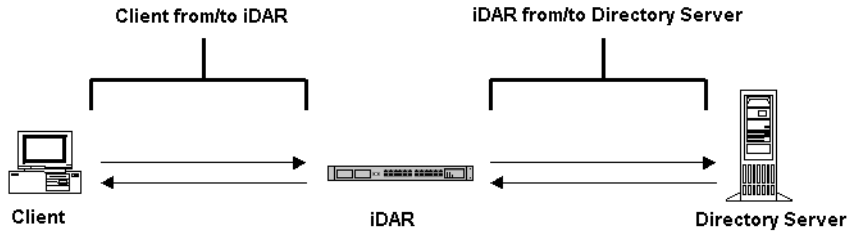
This section explains how to set up iDAR for TLS / SSL-enabled communication with an LDAP client and an LDAP server. It is recommended that you also read the overview part of Chapter 12, “Configuring Security.”

Some of the information in this section is written with an assumption that you are familiar with the basic concepts of public-key cryptography and Secure Sockets Layer (SSL) protocol, and understand the concepts of intranet, extranet, and the Internet security and the role of digital certificates in an enterprise. If you are new to these concepts, we recommend you read the security-related documents available on line at this URL:
`http://docs.iplanet.com/docs/manuals/security.html`

You may also refer to the security-related appendixes of the accompanying manual, *Managing Servers with iPlanet Console.*

iDAR has two separately configurable communication links. Each communication link can be plaintext or encrypted using Transport Layer Security (TLS) or Secure Sockets Layer (SSL) protocol. Availability of the two separate communication links enables you to configure TLS- or SSL-enabled communication between an LDAP client and iDAR and between iDAR and an LDAP directory. Figure 5-1 illustrates this capability of iDAR.

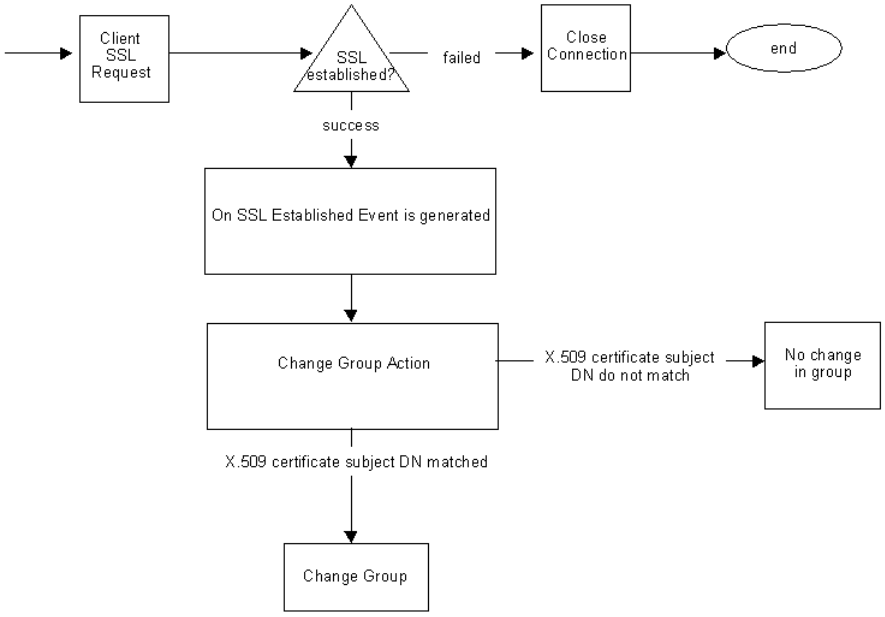
Figure 5-1 Two Separate Communication Links in iDAR



iDAR also supports strong authentication, which is also known as certificate-based authentication. It can verify both client and server certificates, provided the trusted root CA certificate for the certificate being validated is installed and is available to iDAR. Note that the verification process is limited to verifying the validity period and the issuer name of the CA present on the certificate being validated; iDAR does not verify the revocation status of the certificate.

Figure 5-2 illustrates how iDAR can verify the certificates presented to it by the clients after they establish an SSL session.

Figure 5-2 Certificate-Based Authentication of Clients



Note that, unlike some of the SSL-enabled servers that come under the iPlanet Console framework, this version of iDAR is not built with Network Security Services (NSS); for details about NSS, check this site:

`http://www.mozilla.org/projects/security/pki/nss/`

Hence, the iDAR Console doesn't have a built-in Certificate Setup Wizard, which automates the process of requesting and installing an SSL server certificate.

To get an SSL server certificate for iDAR, you must first generate a key pair and a corresponding certificate request using the command-line utility named `certreq`, submit the certificate request to a CA, and obtain a server certificate from the CA.

The `certreq` utility is available in your iDAR installation at this location: `<server-root>/bin/idar/server/bin`

For more information about this utility, see Chapter 12, "Configuring Security."

In general, setting up iDAR for SSL-enabled communication involves these steps:

- Step 1. Install a Server Certificate for iDAR
- Step 2. Set Up SSL Connections Between iDAR and Clients
- Step 3. Set Up SSL Connections Between iDAR and LDAP Servers

NOTE The instructions are written with an assumption that the LDAP clients and backend LDAP servers are set up appropriately and that they own the required certificates.

Step 1. Install a Server Certificate for iDAR

To install a server certificate for iDAR, follow these steps:

- Step A. Generate a Key Pair and a Certificate Request
- Step B. Verify the iDAR Key File and Certificate Request Files
- Step C. Submit the Certificate Request to a CA
- Step D. Copy the Certificate to a Text File
- Step E. Copy the CA Certificate Chain to a Text File

Step A. Generate a Key Pair and a Certificate Request

In this step you use the `certreq` utility to generate a key pair and a corresponding certificate request for iDAR.

1. Open a terminal window in the iDAR host system.
2. Go to this directory: `<server-root>/bin/idar/server/bin`
3. Run the `certreq` command with the appropriate options to generate a key pair and a certificate request for the key pair.

   The command syntax is as follows:

   ```
   certreq -dn <dn> -reqout <filename> -keyout <filename>
   [-dsaparms <filename>] [-bits <bits>]
   ```

   For example, the command to generate an RSA key of size 1024 bits will look like this:

   ```
   ./certreq -dn 'cn=idarhost.siroe.com, o=Siroe Corporation, c=US'
   -reqout idar-certreq.txt -keyout idar-key.txt -bits 1024
   ```

Step B. Verify the iDAR Key File and Certificate Request Files

After you've run the `certreq` command, verify that the key and certificate request files are generated and that they contain the appropriate information.

If you generated a RSA key, the *key file* is the file specified by the `-keyout` parameter. The file should contain the RSA private key. The contents of the file should look similar to the example shown below:

```
-----BEGIN RSA PRIVATE KEY-----

MIICXQIBAAKBgQDrYc78Q9PnU8Q5d0SfFyNXI84sRtGP9NXgP70XxY6Wdg3xoQA
Z/xWyE/0dx0xRR8bzCpi25eTVyYbyQMxY6yu2OxvsywDYwkAN2bRBgePUMVSlx2
Zi62Fr2CzsAaajbOO0yqXFP/gyjXphYorbOvyG78Xp3vFIesWYl6GgzglwIDAQA
BAoGBAOIm1E9N/+/XbMXl0Nmlyn+z2Ch0Vm6gx0kEYduvTXMmiAzwWpKipbRW7V
bwfiqJP1opfe8hBPgD7b8CRo5wQziQlypp7JnFnDjL7U/QARzRATUax/t8iugcb
RdCxQ4PrZ45At/amZpkkWCozYfXA+57LlhW35KxLcstMTNpNu7YPhAkEA+w5fhs
PgUO09eSzMsYyAfvVlctmaLDmcapMLRlNp5AnUYvwK17l0mCrAT

-----END RSA PRIVATE KEY-----
```

CAUTION The information contained in the key file is the base-64 encoding of the RSA private key. Do not make any changes to the file. Also ensure that iDAR has read permission to the file.

The private key file must be protected against theft, tampering, or viewing. You must use the operating system's security feature to restrict access to the file—only you or authorized administrators should have access to this file. It's also important to know if the file is stored on backup tapes or is otherwise available for someone to intercept. Because the destruction of a private key in a disk crash will require you to regenerate the key and reconfigure iDAR, backing up your key file is commensurately important. If you do make copies of the key file, however, you must protect your backups with the same level of security that you use for protecting your original key file.

The *certificate request file* is the file specified by the `-reqout` parameter. This file should contain a certificate request. The contents of the file should look similar to the example shown below:

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

MIIBbTCB1wIBADAuMRMwEQYKCZImiZPyLGQBGRYDY29tMRcwFQYKCZImiZPyLGQB
GRYHaVBsYW5ldDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA62HO/EPT51PE
OXdEnxcjVyPOLEbRj/TV4D+9F8WOlnYN8aEAMWf8VshP9HcdMUUfG8wqYtuXk1cm
G8kDMWOsrtjsb7MsA2MJADdm0QYHj1DFUpcdo2Yutha9gs7AGmo2zjtMqlxT/4Mo
16YWKK2zr8hu/F6d7xSHrFmJehoM4JcCAwEAAaAAMA0GCSqGSIb3DQEBBAUAA4GB
AHcBoLa3Bi3o+HblCIkD6Rx29gShLwVK+QyzrPHrC9iGgrOu

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

The contents of the certificate request file are a base-64 encoding of a value of the ASN.1 type defined in PKCS#10:

```
SIGNED { SEQUENCE {
version Version,
subject Name,
subjectPublicKeyInfo SubjectPublicKeyInfo,
attributes [0] IMPLICIT Attributes } }
```

Do not make any modifications to the file. You will need to send the contents in the certificate request file to the CA, as explained in the next step. (After you receive the certificate from the CA, you may destroy the certificate request file; iDAR does not make use of the certificate request.)

Step C. Submit the Certificate Request to a CA

In this step, you submit the iDAR certificate request to a certificate authority (CA) so that it can sign the certificate request and send the server certificate to you:

1. Open a Web browser window.
2. Navigate to the CA's site, and locate the enrollment form for submitting a SSL server certificate request.
3. In your local file system, open the certificate request file that you created.
4. Copy the certificate request.

   Unless the instructions in the enrollment form specify otherwise, you need to copy the request including the marker lines `-----BEGIN CERTIFICATE REQUEST-----` and `-----END CERTIFICATE REQUEST-----`. The information you copied should look similar to the example below:

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

MIIBbTCB1wIBADAuMRMwEQYKCZImiZPyLGQBGRYDY29tMRcwFQYKCZImiZPyLGQB
GRYHaVBsYW5ldDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA62HO/EPT51PE
OXdEnxcjVyPOLEbRj/TV4D+9F8WOlnYN8aEAMWf8VshP9HcdMUUfG8wqYtuXk1cm
G8kDMWOsrtjsb7MsA2MJADdm0QYHj1DFUpcdo2Yutha9gs7AGmo2zjtMqlxT/4Mo
16YWKK2zr8hu/F6d7xSHrFmJehoM4JcCAwEAAaAAMA0GCSqGSIb3DQEBBAUAA4GB
AHcBoLa3Bi3o+HblCIkD6Rx29gShLwVK+QyzrPHrC9iGgrOu

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

5. Go back to the CA's enrollment form, and paste the certificate request into the text area provided in the form.
6. Fill out any other information required by the CA; some CA's mandate this in order to approve a certificate request.
7. Submit the request.

   If you submitted the request to a third-party or public CA, as opposed to an internally-deployed CA, you might want to keep the response you receive from the CA. It will contain information, for example, a reference number, that will help you to track your request.

   The issuance or availability of your certificate will depend on the CA's policy for issuing server certificates. Some CAs might take longer time to process your request and issue the certificate. You should wait till you get the certificate and then continue with the remaining instructions.

Step D. Copy the Certificate to a Text File

When you get a notification about the availability of the server certificate, follow the instructions in this step:

1. In the notification you got from the CA, locate the server certificate in the base-64 encoded format.
2. Copy the certificate.

   You need to copy the certificate including the marker lines `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----`. The information you copied should look similar to the example below:

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

MIIBbTCB1wIBADAuMRMwEQYKCZImiZPyLGQBGRYDY29tMRcwFQYKCZImiZPy
LGQBGRYHaVBsYW5ldDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA62HO
/EPT51PEOXdEnxcjVyPOLEbjgTV4D+9F8WOlnYN8aEAMWf8VshP9HcdMUUfG
wqYtuXk1cmG8kDMWOsrtjsb7MsA2MJADdm0QYHj1DFUpcdo2Yutha9gs7AGm
o2zjtMqlxT/4Mo16YWKK2zr8hu/F6d7xSHrFmJehoM4JcCAwEAAaAAMA0GCS
qGSIb3DQEBBAUAA4GBAHcBoLa3Bi3o+HblCIkD6Rx29gShL
-----END CERTIFICATE-----
```

3. Create a text file, for example, `idar-cert.txt`.
4. Paste the certificate you copied to the text file and save.
5. Put the file in a location that's accessible and readable by iDAR. Also note the file path. You will be required to specify the path later.

Step E. Copy the CA Certificate Chain to a Text File

If you set up iDAR for SSL-client authenticated communication with LDAP servers or SSL-server authenticated communication with LDAP clients, those applications will need to trust the CA that signed the iDAR's server certificate. (If this CA is subordinate to another trusted root CA, then the applications will need to trust the root CA.)

To establish trust between the said applications and iDAR, you (or the administrators of those applications) will need to install the CA certificate chain in appropriate LDAP clients' and LDAP servers' certificate databases.

The instructions in this step will help you save the CA certificate chain to a file for later use.

1. Locate the CA certificate and / or CA certificate chain in its base-64 encoded format.

   Some CAs send it along with the certificate, where as some CAs might make it available on the CA site. Check the notification you received from the CA when you got the server certificate.
2. Create a text file, for example, `idar-cachain.txt`.
3. Copy the root certificate to the text file and save.
4. If required, send the file to LDAP client and server administrators.

Step 2. Set Up SSL Connections Between iDAR and Clients

To set up SSL connections between iDAR and LDAP clients, follow these steps:

- Step A. Create a File with CA Certificates in PEM Format
- Step B. Add iDAR CA Certificate to Clients' Trust Databases
- Step C. Make Changes to the iDAR System Configuration
- Step D. Make Changes to the iDAR Network Groups

Step A. Create a File with CA Certificates in PEM Format

This step is required if you want iDAR to verify the certificates presented to it by the clients after they establish an SSL session (see Figure 5-2 on page 70).

1. Create a text file.
2. Copy the trusted root certificates of CAs that iDAR can trust.

   The certificates must be in the Privacy Enhanced Message (PEM) format, including the markers as shown in the example below:

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

   MIIDkjCCAnqgAwIBAgIBADANBgkqhkiG9w0BAQUFADCBgzELMAkGA1UEBhMCVVM
   xCzAJBgNVBAgTAkRDMRMwEQYDVQQHEwpXYXNoaW5ndG9uMRcwFQYDVQQKEw5BQk
   EuRUNPTSwgSW5jLjEZMBcGA1UEAxMQQUJBLkVDT00gUm9vdCBDQTEeMBwGCSqGS
   Ib3DQEJARYPa2RhZ3Vpb0BhYmEuY29tMB4XDTk4MDcyOTE2NTk1MloXDTA1MDcy
   NzE2NTk1MlowgYMxCzAJBgNVBAYTAlVTMQswCQYDVQQIEwJEQzETMBEGA1UEBxM
   VFzaGluZ3RvbjEXMBUGA1UEChMOQUJBLkVDT00sIEluYy4xGTAXBgNVBMTEEFCQ
   S5FQ09NIFJvb3QgQ0ExHjAcBgkqhkiG9w0BCQEWD2tkYWd1aW9AYWJhLmNvbTCC
   ASIwDQYJKoZIhvcNAQEBBhuruhtAQoCggEBMae3L3cDgkaUcaSm5lrjGmJvhvFo
   hFOhGYNmfH/H5mhM9a0kouli57Wp5EybSBGp6HUP9zVqdtEFsIE6asCKkaIHIaD
   zN0sVixVm81Nj0zXpPjmgK1obfxzEFNQ3Xo

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

   For more information, check the sample file available at this location: `<server-root>/idar-<hostname>/etc/rootcerts.pem`
3. Save the file.

   This file must contain all the CA certificates in the PEM format.
4. Make sure iDAR has read permission to the file. Also note the file path as you'll be required to specify this information later.

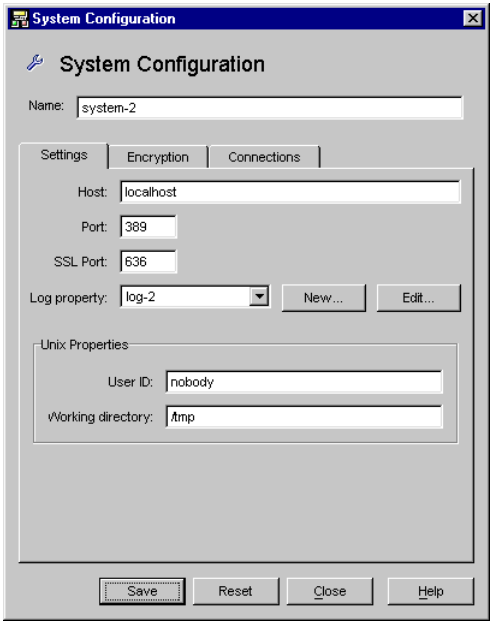
Step B. Add iDAR CA Certificate to Clients' Trust Databases

When iDAR presents its certificate to an LDAP client, the client tries to verify the validity of the certificate. As a part of this verification process, the client checks whether the CA that has issued the certificate is trusted by the client. For this reason, the root certificate of the CA that issued iDAR's server certificate must be installed in the client's trust database.

In the last step of installing iDAR's server certificate, you copied the iDAR's CA certificate to a text file. Follow the documentation for each client application, and install the CA certificate in its trust database.

Step C. Make Changes to the iDAR System Configuration

The Settings and Encryption tabs in the System Configuration window enable you to define SSL-enabled communication criteria for iDAR. For details, see “Creating System Configuration Objects,” on page 63.



Make the following changes to the appropriate system configuration objects, and save your changes.

- In the Settings tab, specify a value in the “SSL port” field. iDAR will listen on the port number you specify for LDAPS (LDAP over SSL) connections. By default, iDAR does not listen for connections from LDAPS clients. This value must be present to enable LDAPS connections from clients that use the alternative port 636 method to establish TLS/SSL. This value must be different from the value in the Port field. (This option also requires TLS/SSL configuration found on the Encryption tab.)

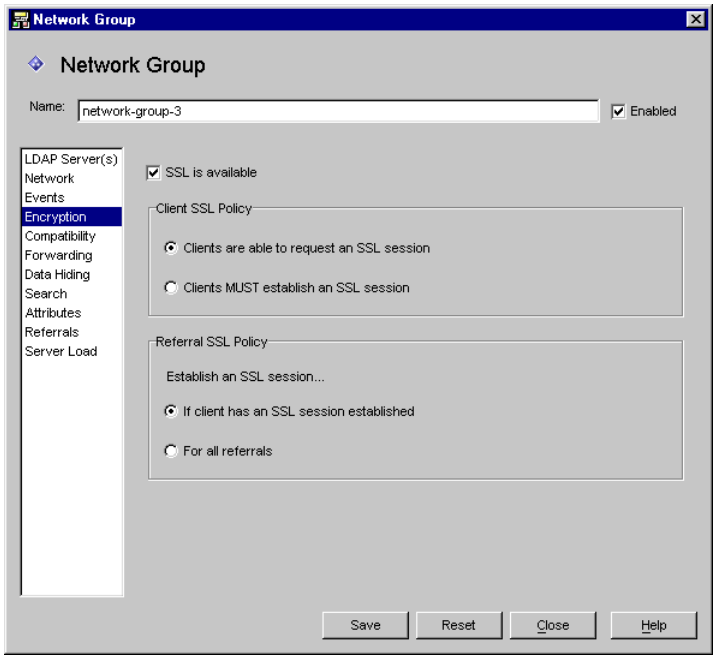
  If you need description for the parameters, click the Help button or check Step 5 on page 65 .

- In the Encryption tab, specify all the required information.

  If you need description for the parameters, click the Help button or check Step 6 on page 65 of Creating System Configuration Objects.

Step D. Make Changes to the iDAR Network Groups

iDAR uses network groups to identify clients and determine their access privileges to the information contained in an LDAP directory; for details, see Chapter 6, “Creating and Managing Groups.”



In each group that you’ve configured, set the appropriate options in the Encryption tab to indicate whether you want to force the client to start a TLS session before sending any LDAP operation, leave the decision to the client, or disallow the client to start a TLS session. For example, you might want to enable “SSL is available” and “Clients MUST establish an SSL session” options. For more information about the options presented in the Encryption tab, see Step 9 on page 92 of Chapter 6, “Creating and Managing Groups.”

If referral following is enabled, you should check the Referral SSL Policy. Referral following is enabled by selecting Referrals in the list on the left side of the window.

Step 3. Set Up SSL Connections Between iDAR and LDAP Servers

To set up SSL connections between iDAR and LDAP servers, follow these steps:

- Step A. Create a File With CA Certificates in PEM Format
- Step B. Add iDAR CA Certificate to the LDAP Servers' Trust Databases
- Step C. Make Changes to the LDAP Server Properties

Step A. Create a File With CA Certificates in PEM Format

This step is required if you want iDAR to verify the certificate presented to it by an LDAP server. If you've already created a file with the trusted CA certificates of the CAs that issued the LDAP client certificates, add the CA certificates of the LDAP servers to the same file. For details, see "Step A. Create a File with CA Certificates in PEM Format," on page 76.

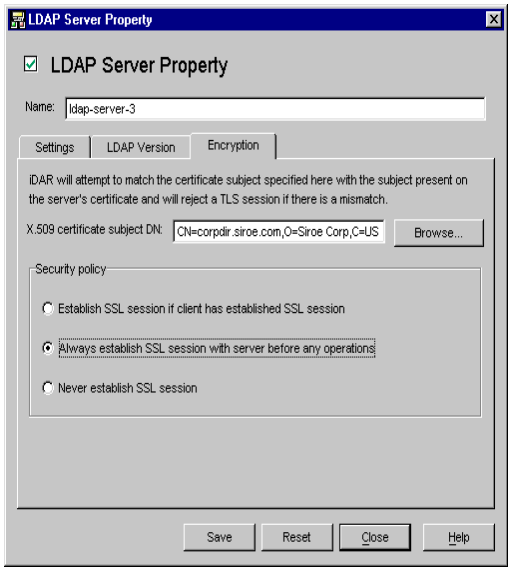
Step B. Add iDAR CA Certificate to the LDAP Servers' Trust Databases

When iDAR presents its certificate to an LDAP server, the server tries to verify the validity of the certificate. As a part of this verification process, the server checks whether the CA that has issued iDAR's certificate is trusted by the server. For this reason, the root certificate of the CA that issued iDAR's server certificate must be installed in the LDAP server's trust database.

In the last step of installing iDAR's server certificate, you copied the iDAR's CA certificate to a text file. Follow the documentation for each LDAP server, and install the CA certificate in its trust database. If you're using iPlanet Directory Server, you can use the Manage Certificates Wizard, which can be launched from the Tasks tab of the Directory Server Console, to add the CA certificate to the Directory Server's trust database.

Step C. Make Changes to the LDAP Server Properties

The Encryption tab in the LDAP Server Property window enables you to define SSL-enabled communication criteria for each LDAP server. For details, see "Creating LDAP Server Property Objects," on page 121.



Make the following changes to the appropriate LDAP Server property objects, and save your changes.

- Set the “Security policy” option to an appropriate value so that iDAR will always establish SSL/TLS to the backend server, never establish TLS/SSL to the backend server, or only establish SSL/TLS with the backend server when the client does the same to iDAR.
- Set the “X.509 certificate subject DN” field to the LDAP server’s certificate subject name (the subject attribute in the X.509 certificate). If specified, iDAR will attempt to match the certificate subject with the subject present on the LDAP server’s certificate and will reject a TLS session if there is a mismatch. (This attribute allows iDAR to authenticate the LDAP server to which it is connecting. iDAR accepts any name if this attribute is not set.)

Chapter 6

Creating and Managing Groups

When an LDAP client requests a service from an LDAP directory, it connects to iPlanet Directory Access Router (iDAR), which in turn identifies the client, determines whether the client is allowed to request the service from the directory, imposes configured restrictions, and then forwards the request to the appropriate directory. This chapter explains how to configure iDAR to identify clients and impose any restrictions using the iDAR Configuration Editor Console.

The chapter has the following sections.

Overview of Groups

iDAR network groups are key to understanding how iDAR works—they define how iDAR should identify an LDAP client and what restrictions iDAR should enforce on clients that match that group. It's important that you understand iDAR groups clearly in order to use them to effectively control directory access by LDAP clients.

You use network groups to identify the following:

- A client
- A set of LDAP directories to which iDAR can forward requests from a client.

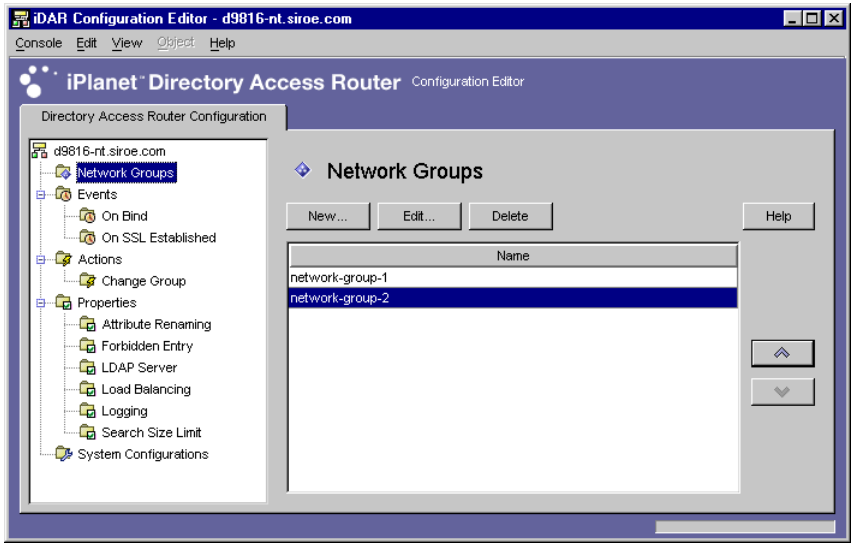
- A set of operations a client can perform while interacting with its set of directories.
- The data accessible to a client while interacting with its set of directories. (Because iDAR enables you to hide certain entries and rename attributes in a directory, you can effectively control which data contained within a directory is viewable by a client.)

iDAR determines the group membership for a client by attempting to match the connection's origination attributes with a group's criteria. The server checks currently-configured groups in the descending order of priority, from the highest to the lowest priority. The first network group criteria to match the connection's origination attributes receives the connection. For this reason, it's important to create separate groups for generic and specific criteria, and prioritize the groups from most specific to most general.

If no groups are found to match a client, the client's request is rejected and the connection is closed. For this reason, there must be at least one group entry in the iDAR configuration.

The order of priority for groups is specified by their placement in the Network Groups window of the iDAR Configuration Editor Console (see Figure 6-1). In this window, groups on the bottom of the list have less priority than those towards the top. The order of evaluation of groups with equal priority is undefined.

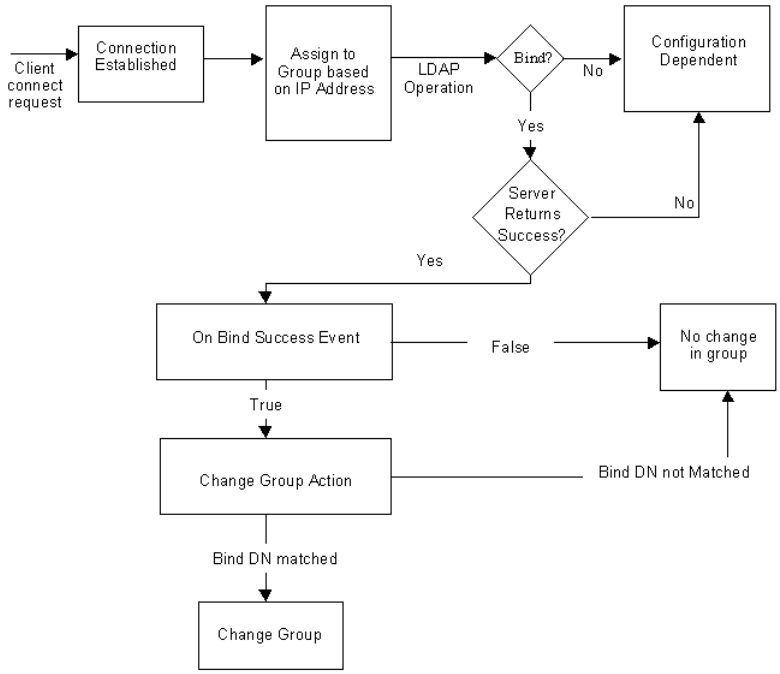
Figure 6-1 iDAR Configuration Editor Console: Network Groups Window



Note that clients are initially identified into a group based on the network address they connect from, for example, their IP address and/or domain name. They may change their group after a successful bind; for details, see Chapter 8, “Creating and Managing Event Objects.” Once a client obtains membership in a group, it implies that all the properties of the group apply to the client.

Figure 6-2 illustrates how groups are evaluated by iDAR in response to a client query.

Figure 6-2 iDAR Decision Tree for Determining Group Membership



Network criteria for groups can be based on the following:

- IP address or network mask of the hosts
  - Single IP address (for example, `129.153.129.14`)
  - IP quad/match bits (for example, `129.153.129.0/24`)
  - IP quad/match quad (for example, `129.153.129.0/255.255.255.128`)

- Domain name of the hosts
  - Full name (for example, `siroe.com`)
  - Suffix name (for example, `eng.siroe.com`)
- Special
  - `ALL` (This is to be used for “catch-all” groups.)
  - `0.0.0.0` (This is to be used for groups to which initial membership is not considered, for example, if a group is only used for clients to switch to when they bind.)

To further understand how iDAR evaluates groups, take a look at the sample groups listed in Table 6-1. It shows five groups, created with specific to generic network criteria, and listed in the descending order of priority.

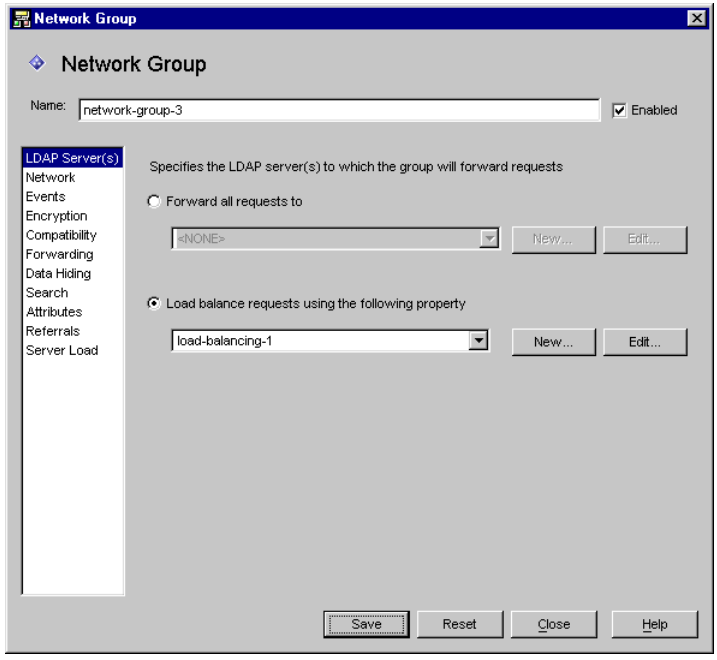
Table 6-1 Sample Groups

Priority	Group Name	Network Criteria
5	Admin-machine	129.153.129.72
4	IT-management-subnet	129.153.120.0/24
3	Operations	*.ops.siroe.com
2	Catch-all	ALL
1	Trusted	0.0.0.0

When an LDAP client requests a service from an LDAP directory, iDAR checks whether the request is from IP address `129.153.129.72`. If it isn’t, iDAR checks whether the request matches `129.153.129.0/24`. If it doesn’t, iDAR checks whether the request originated from `*.ops.siroe.com`. If it didn’t, iDAR places the connection in a `Catch-all` group, and then moves to the next step in the decision tree (see Figure 6-2).

Figure 6-3 shows that part of the iDAR Configuration Editor Console where you are able to create groups.

Figure 6-3 iDAR Network Group Definition



Notice that when creating a network group, you're given the opportunity to specify a combination of criteria. Table 6-2 summarizes them.

Table 6-2 List of Available Criteria for Network Groups

Criteria	Description
LDAP Server(s)	Enables you to specify the LDAP servers to which requests from clients in a network group should be forwarded. Shows the list of existing objects for the LDAP Server and Load Balancing properties; for details, see "LDAP Server Property," on page 121 and "Load Balancing Property," on page 126. The iDAR configuration must include a group with either of the two properties. Otherwise, iDAR will fail to start.
Network	Enables you to specify connection details and other network criteria for clients so that their requests get sorted or filtered into the appropriate groups.

Table 6-2 List of Available Criteria for Network Groups *(Continued)*

Criteria	Description
Events	Enables you to specify which events, if any, to associate with a group, so that clients in the group can effectively change group after binding successfully to a specified directory. Shows the list of existing objects for events; for details, see "Creating Groups," on page 87.
Encryption	Enables you to specify encryption criteria for the group (for example, to specify whether clients can request an SSL session).
Compatibility	The LDAP v2 specification (RFC 1777) does not allow a client to bind multiple times in one session. However, some clients expect this functionality. This option can be set to interoperate with these clients.
Forwarding	Enables you to specify the criteria for passing the bind, compare, and other LDAP requests to the server.
Data Hiding	Enables you to specify which subtree, entries, or attributes of the entries in a directory are to be hidden from a group. Shows the list of existing objects for the Forbidden Entry property; for details, see "Forbidden Entry Property," on page 117.
Search	Enables you to specify the scope and size limit of searches for a group. Shows the list of existing objects for the Search Size Limit property; for details, see "Search Size Limit Property," on page 130.
Attributes	Enables you to specify rules for preventing certain kinds of search and compare operations from reaching the LDAP server. Shows the list of existing objects for the Attribute Renaming property; for details, see "Attribute Renaming Property," on page 114.
Referrals	Enables you to specify whether a group should forward, follow, or discard referrals returned by the server. Note that a client that does not implement LDAPv3 will not understand forwarded referrals. This setting applies to all referrals except for the search-continuation referrals.
Server Load	Enables you to specify details such as the total number of connections to a group, simultaneous and total operations per connection, simultaneous operations per IP address, and so on.

Creating Groups

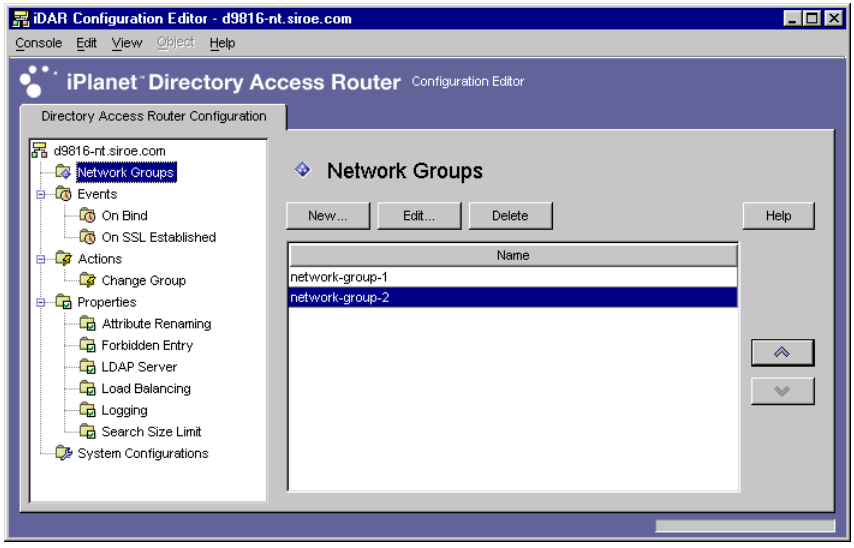
This section explains how to create groups using the iDAR Configuration Editor Console. Before you start creating a group, be sure to read section “Overview of Groups,” on page 81 and understand the significance of iDAR groups. After you create the required groups and prioritize them, be sure to test the configuration to see if the groups filter client requests as desired.

Notice that when creating a network group, you’re given the opportunity to specify a variety of criteria. The instructions provided in this section present all these criteria in the order in which they appear on the UI, and rely on your judgement to set the appropriate criteria for a group.

To create a network group in iDAR, follow these steps:

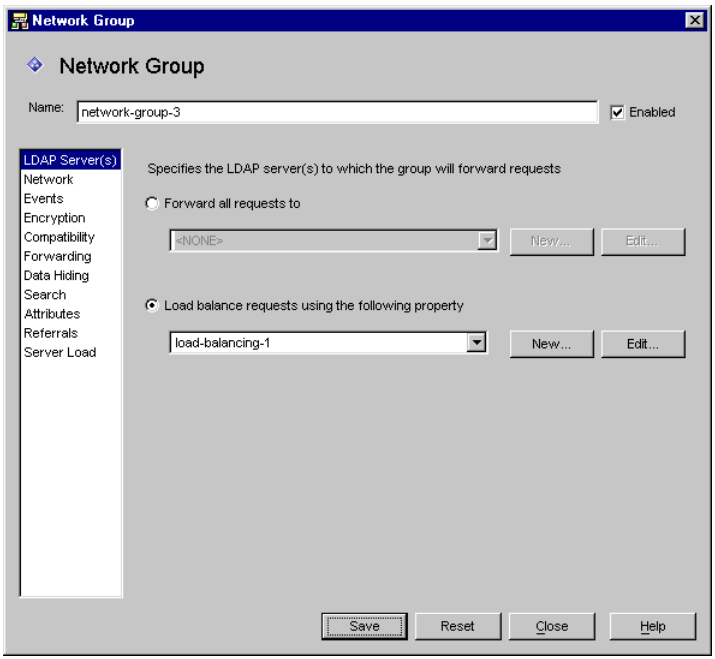
1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, select Network Groups.

   The right pane shows the list of existing groups.



3. Click New.

   The Network Group window appears.



4. In the Name field, type a name for the group. The name must be a unique alphanumeric string. (This value must be present as it forms the RDN of entries of this class.)

5. Make sure that the Enabled option is selected; by default, it is selected. For a group to be part of an iDAR configuration, this option must be selected. Deselect the option to disable the group in a configuration.

6. If you want the group to forward requests to LDAP servers, make sure that LDAP Server(s) is selected on the left frame, and specify the appropriate values on the right frame.

   The description of the on-screen elements is as follows:

   Forward all requests to. Select this option if you want the group to forward requests to a specified LDAP server. The associated drop-down list shows existing objects for the LDAP Server property; see “LDAP Server Property,” on page 121. Select an appropriate object. By default, no (<NONE>) objects are selected. If there isn’t an object, you can create one on the fly by clicking on the New button.

   New. Displays a dialog to create a new object for the LDAP Server property.

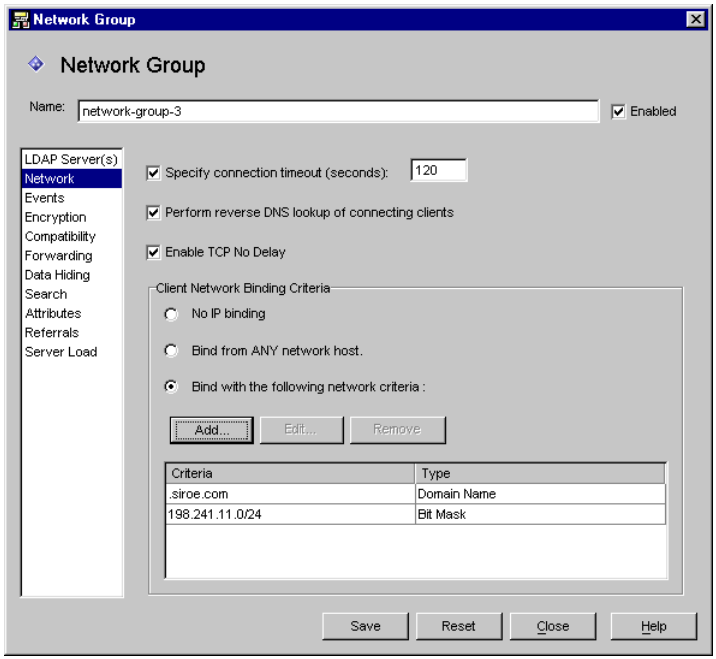
   Edit. Displays a dialog to edit an existing LDAP Server property.

Load balance requests using the following property. Select this option if you want the group to use a Load Balance property to handle requests from clients. The associated drop-down list shows existing objects for the Load Balance property; see “Load Balancing Property,” on page 126. Select an appropriate object. By default, no (<NONE>) objects are selected. If there isn’t an object, you can create one on the fly by clicking on the New button.

New. Displays a dialog to create a new Load Balance property.

Edit. Displays a dialog to edit an existing Load Balance property.

7. If you want to specify network criteria for the group to sort or filter requests, select Network on the left frame and specify the appropriate values on the right frame.



The description of the on-screen elements is as follows:

Specify connection timeout. Select this box if you want to enter a period of client inactivity after which iDAR may close the connection to the client. The value must be a number in seconds, typically 120 or more. By default, no value is present, which also means to not timeout connections. Note that if TCP keepalives are not enabled, this attribute must be present to keep iDAR from being clogged by lost client connections.

Perform reverse DNS lookup of connecting clients. By default, this option is enabled. If Reverse DNS lookup is disabled, iDAR will not perform a reverse DNS lookup to find the domain name of the connecting client. Disabling Reverse DNS lookup can sometimes significantly improve iDAR performance. If you have used a domain name or a domain name suffix as a value in the "Client Network Binding Criteria," you must not disable Reverse DNS lookup, otherwise iDAR will not function properly.

Enable TCP no delay. By default, this option is enabled. If the option is disabled, then iDAR will disable the Nagle Algorithm for connections between itself and clients that fall into this group. "TCP no delay" should only be disabled if the network bandwidth between iDAR and clients is small; however, it may create substantial performance degradation.

Client Network Binding Criteria. Use this section to specify which clients are able to bind in this network group.

No IP binding. Select this option if clients are only to switch when they bind to the group. By default, this option is enabled. Disable the option if the group is only used for clients to switch to when they bind.

Bind from ANY network host. Select this option if all hosts are allowed to bind with this network group.

Bind with the following criteria. Select this option to specify the domain names or IP addresses of the hosts that match the network group; in this case, the group must specify the domain name or IP address of the host that will bind to it.

Add. Displays a dialog to add a network criteria. There are four options: "Domain Name," "IP address," "IP address and bits," and "IP address and quad."

Edit. Displays a dialog to edit a network criteria.

Remove. Displays a dialog to remove a network criteria.

Domain name dialog. Specify the domain name of a client that can bind to a network group, for example, `foo.siroe.com`. Note that iDAR does not assume any domain suffix by default; hence, complete domain names must be provided. A domain name suffix with a leading period, for example, `.siroe.com` will cause all hosts with domain names that end in that suffix to match.

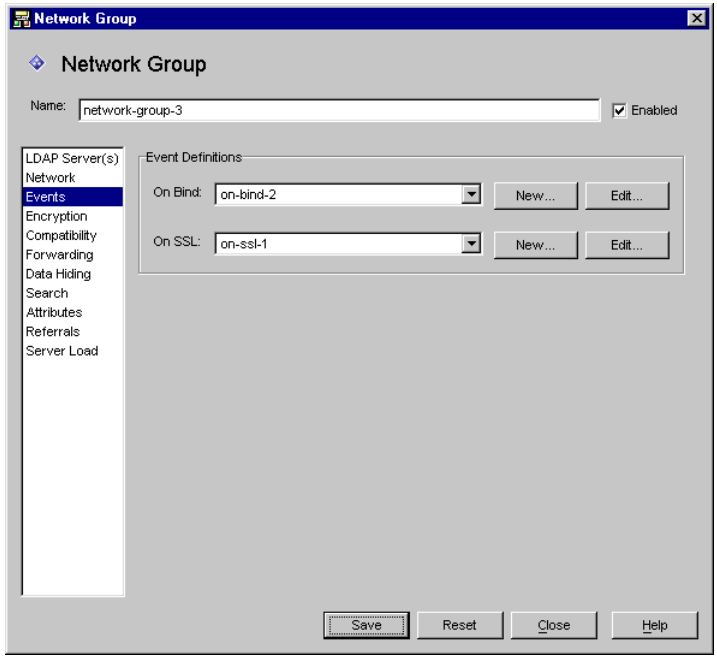
IP address. Specify a single IP address in dotted decimal form, for example, `198.214.11.1`.

IP address and bits. Specify an IP network mask, in the form of `<network number>/<mask bits>`, for example, `198.241.11.0/24`. The first half is the network number and the second half indicates the number of bits of the network number necessary for matching.

IP address and quad. Specify an IP network mask, in the form of a pair of dotted decimal quads, for example, `198.241.11.0/255.255.255.128`. The first half is a network number, the second half indicates the bits of the network number necessary for matching. For example, `198.214.11.0/255.255.255.128` will match a host with IP address `198.214.11.63` but not the one with IP address `198.214.11.191`.

Note that use of domain names or domain name suffixes requires “Perform reverse DNS lookup of connecting client” to be enabled.

8. If you want to associate an event-driven action with the group (for example, to change clients from one group to another), select Events on the left frame and specify the appropriate values on the right frame.



The description of the on-screen elements is as follows:

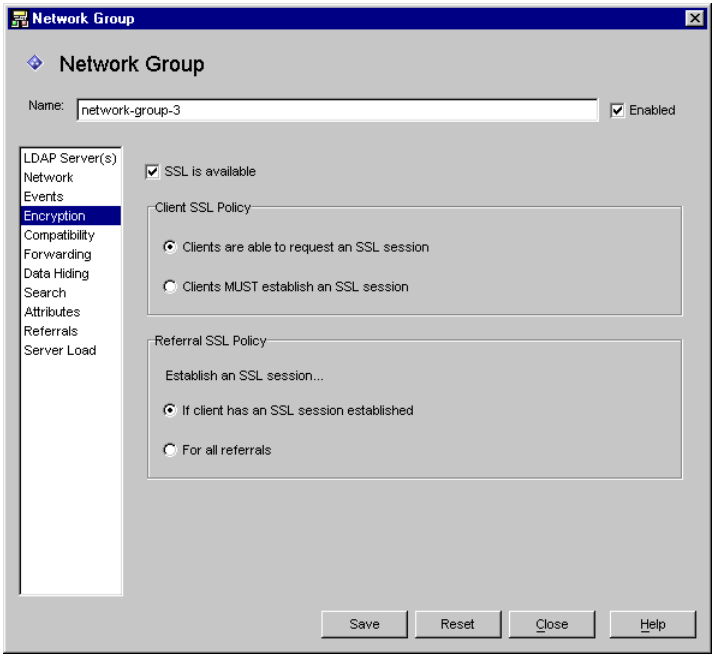
On bind. The drop-down list shows existing objects for OnBindSuccess events; see “Creating OnBindSuccess Event Objects,” on page 138. Select the name of an object that will be performed when a client successfully completes a bind operation. By default, no (<NONE>) objects are selected. If there isn’t an object, you can create one on the fly by clicking on the New button.

On SSL. The drop-down list shows existing objects for OnSSLEstablished events; see “Creating OnSSLEstablished Event Objects,” on page 141. Select the name of an object that will be performed when a client successfully establishes an SSL session. If there isn’t an object, you can create one on the fly by clicking on the New button.

Edit. Displays a dialog box for editing the behavior of an event.

New. Displays a dialog box for creating a new event.

9. If you want to specify encryption criteria for the group (for example, to specify whether clients can request an SSL session), select Encryption on the left frame and specify the appropriate values on the right frame.



The description of the on-screen elements is as follows:

SSL is available. Select this option if Client SSL and Referral SSL is available to the group. By default, Client SSL and Referral SSL is unavailable. iDAR will not permit a Client SSL or Referral SSL session if this option is disabled. This is the default.

Clients are able to request an SSL session. Select this option if the clients in the group will establish an SSL session provided that a client request SSL.

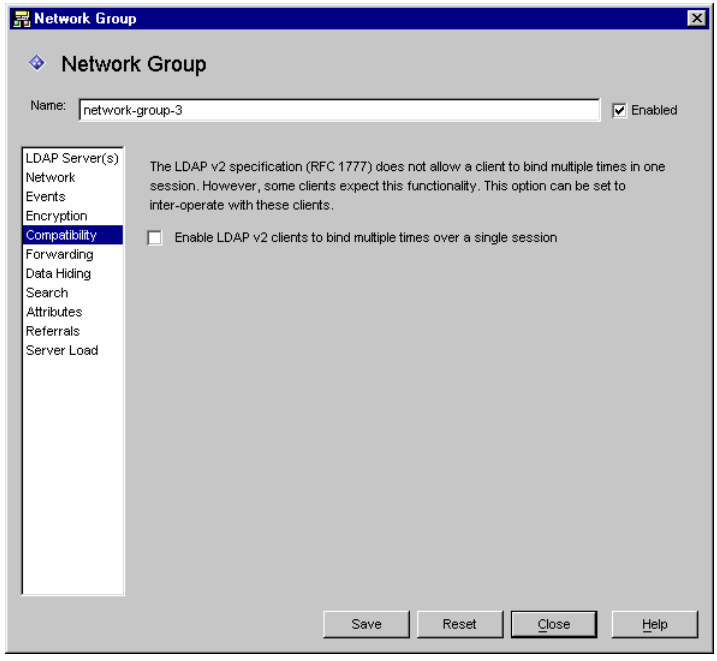
Clients MUST establish an SSL session. Select this option if the clients in the group must establish an SSL session before performing any operation.

Referral SSL policy. Configure the SSL policy while following referrals.

If client has an SSL session established. If "SSL is available" is enabled and if this option is enabled, iDAR will only initiate SSL for clients in that group if the client already has an SSL session established with iDAR.

For all referrals. Enable this option, if, upon a referral, iDAR will initiate an SSL session before the operation is forwarded.

10. If you want to specify compatibility criteria for the group (for example, to allow a client to bind multiple times in one session), select Compatibility on the left frame and specify the appropriate values on the right frame.



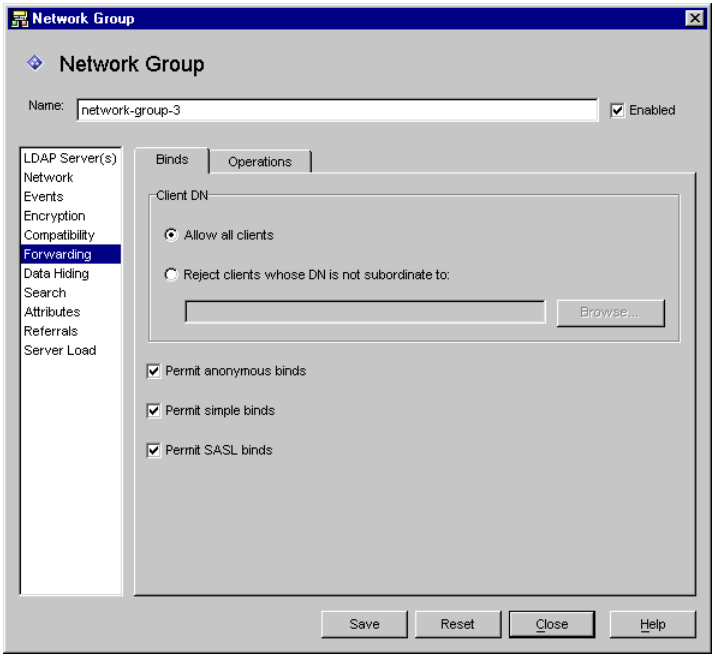
The description of the on-screen element is as follows:

Enable LDAP v2 clients to bind multiple times over a single session. The LDAP v2 specification (RFC 1777) does not allow a client to bind multiple times in one session. However, some clients expect this functionality. Select this if you want the group to allow a client to bind multiple times in one session.

11. If you want to specify request-forwarding criteria for the group, select Request Forwarding on the left frame and specify the appropriate values on the right frame.

Once iDAR has accepted a connection from the client and matched a group, it will wait for the client to send the LDAP operation. iDAR uses the “Client DN,” “Permit Anonymous binds,” “Permit simple binds,” and “Permit SASL binds” to determine whether to pass the bind request to the server, or reject the bind request and close the client’s connection.

If the client’s bind passes enabled tests, iDAR will forward it to the server. If the server accepts the bind, the connection is established. If, however, the server returns an error indication for the bind request, iDAR will forward the error indication to the client, and then close the connection to the client, if the client was using LDAPv2.



The description of the elements in the Binds tab is as follows:

Allow all clients. By default, this option is enabled, which permits access by all clients.

Reject clients whose DN is not subordinate to. Select this option if you want the group to check for a distinguished name (DN). Any client that provides a distinguished name in its bind that is not subordinate to a the specified DN will be rejected. Use the Browse button to browse an LDAP directory in order to construct a DN.

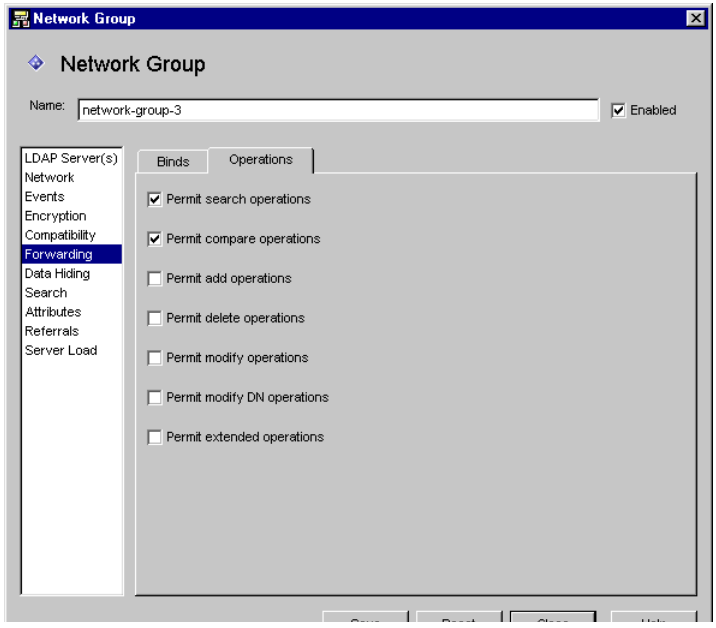
Permit anonymous binds. By default, this option is enabled, which permits a bind even if a client has not supplied a password. Disable the option to forbid anonymous binds.

Permit simple binds. By default, this option is enabled, which permits a client to supply a password in the clear. Disable the option to forbid clear text password authenticated bind requests.

Permit SASL binds. By default, this option is enabled, which specifies that SASL binds are permitted. Disable the option to forbid SASL authentication.

12. Select the Operations tab and specify which operations are to be forwarded.

    iDAR by default forwards search and compare requests. iDAR also recognizes an unbind request and closes the connection to the LDAP server.



The description of the elements in the Operations tab is as follows:

Permit search operations. By default, this option is enabled. Disable the option to prevent iDAR from forwarding search requests to the server.

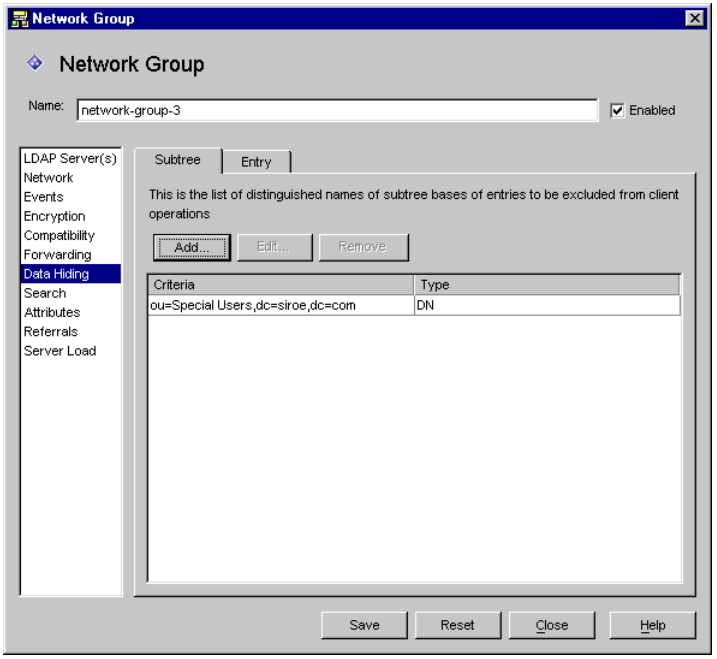
Permit compare operations. By default, this option is enabled. Disable the option to prevent iDAR from forwarding compare requests to the server.

Permit add, delete, modify, modify DN, and extended operations. By default, iDAR does not forward add, modify, delete, modify DN, or extended operations requests. To permit forwarding of these operations, enable the appropriate operation to be allowed.

Note that you must enable “Permit extended operations” if you want your clients to be able to negotiate Start TLS.

13. If you want to specify data hiding criteria for the group, select Data Hiding on the left frame and specify the appropriate values on the right frame.

    Use the Subtree tab to specify which part of the directory tree is to be hidden and Entry tab to specify entries or attributes to be hidden.



The description of the elements in the Subtree tab is as follows:

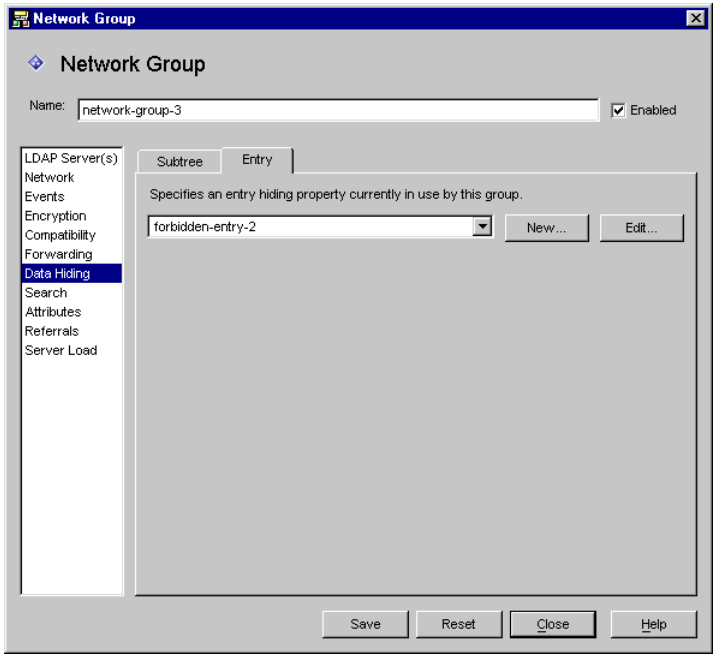
Hiding a subtree of entries. Operations that request entries at or below a forbidden subtree will be rejected with an insufficient access error. Entries that match a search filter and are inside a forbidden subtree are dropped. Note that this option does not remove DN syntax attributes whose values fall under the subtree from entries that are being returned as part of the result.

Add. Displays a dialog box to add a distinguished name to a list of the base of a subtree of entries to be excluded. The default, if distinguished names are not present in a network group, is to allow access to all entries in the directory. An entry in the list has dn syntax.

Edit. Displays a dialog box to edit a distinguished name.

Remove. Removes a distinguished name from the list.

14. Select the Entry tab and specify which entries or attributes are to be hidden.



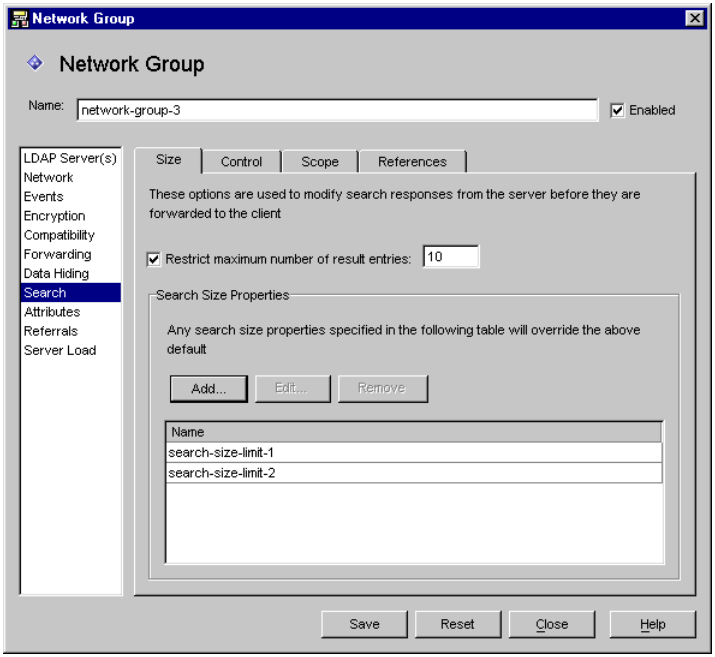
The description of the elements in the Entry tab is as follows:

Specifies an entry hiding property currently in use by this group. The drop-down list shows existing objects for the Forbidden Entry property; see “Forbidden Entry Property,” on page 117. Select the name of an object. By default, no (<NONE>) objects are selected. If there isn’t an object, you can create one on the fly by clicking on the New button.

New. Displays a dialog to create a new Forbidden Entry property.

Edit. Displays a dialog to edit an existing Forbidden Entry property.

15. If you want to specify search attributes for the group, select Search on the left frame and specify the appropriate values on the right frame.



The description of the elements in the Size tab is as follows:

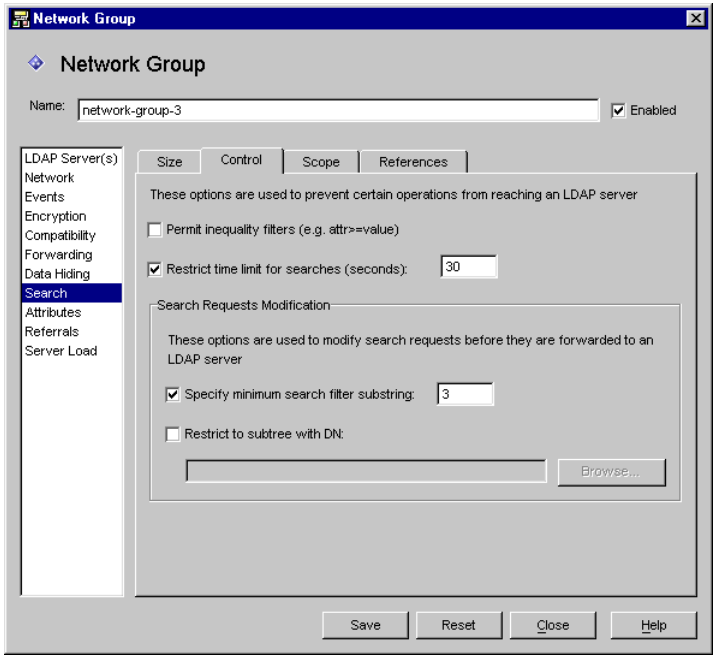
Restrict maximum number of result entries. Enable this option to specify the maximum number of result entries that may be returned to a client at one time from a single search operation. The value may be any number greater than zero, and if reached, will cause an `administrativeLimitExceeded` error to be indicated to the client and subsequent entries will be discarded. The default, if this property is disabled, is to not discard entries.

Add. Displays a dialog to add a Search Size Limit property. For details, see “Search Size Limit Property,” on page 130.

Edit. Displays a dialog to edit a Search Size Limit property.

Remove. Displays a dialog to remove a Search Size Limit property. (This action removes the property from the group without displaying a dialog.)

16. Select the Control tab and specify the criteria for controlling search filters.



The description of the elements in the Control tab is as follows:

Permit inequality filters. By default, this option is enabled. Permit inequality filters specifies whether clients are permitted to request searches that contain inequality filters (`attr>=value`) and (`attr<=value`). Disable this option if a network group does not permit inequality searches to be performed.

Restrict time limit for searches. Enable this option and enter a value in seconds for a network group to specify a maximum time limit in seconds for search operations. If the client specifies a time limit that is larger than the value given in this option, the value specified for this network group will override the client’s request. By default, this option is disabled and a network group will allow the client to set any time limit, including no limit.

Specify minimum search filter substring. Enable this option and enter a value to specify the minimum permissible length of a substring in a search filter. The value is a number greater than one. The default, if this option is disabled, is to allow any size of substring in a search filter. This option should be enabled in the a network group if you wish to restrict the kinds of searches that may be performed by web robots. For example, a value of 2 will block searches like (`cn=A*`).

Restrict to subtree with DN. Enable this option and specify the base of a subtree for all operations. This option has dn syntax. If this option is disabled, then there is no restriction to a minimum base.

Operations whose target entry is at or below the minimum base entry are not affected by this option. If the target entry is superior to the minimum base entry, and the operation is a subtree search, then the query will be rewritten before being sent to the server, to change the target entry to be the minimum base. If the target entry is not below the minimum base or a superior of it, the request will be rejected with a no such object error.

For example, if the “Restrict to subtree with DN” is set as:

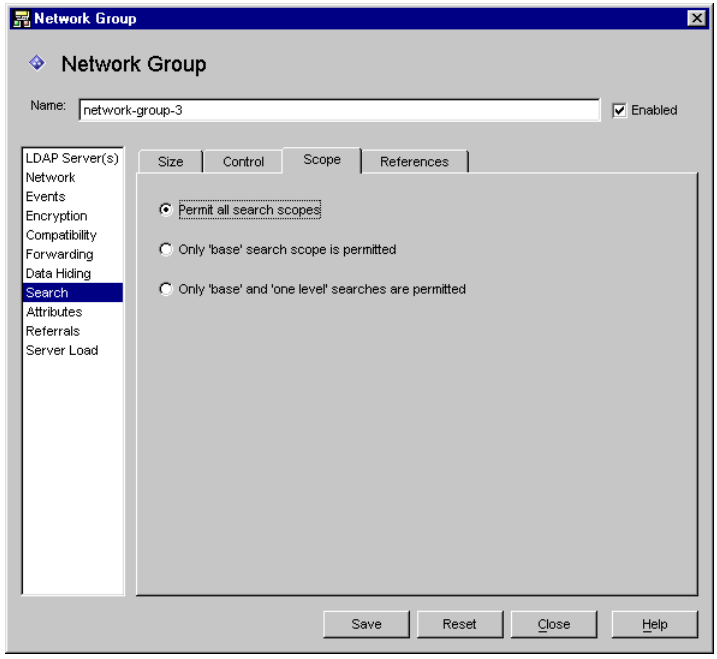
```
o=Siroe, st=California, c=US
```

and a subtree search of `st=California, c=US` is received, the search will be rewritten such that the server performs a subtree search of

```
o=Siroe, st=California, c=US
```

Browse. Displays a dialog to aid in constructing a valid DN.

17. Select the Scope tab and specify the search scope (that a client may specify in a search request).



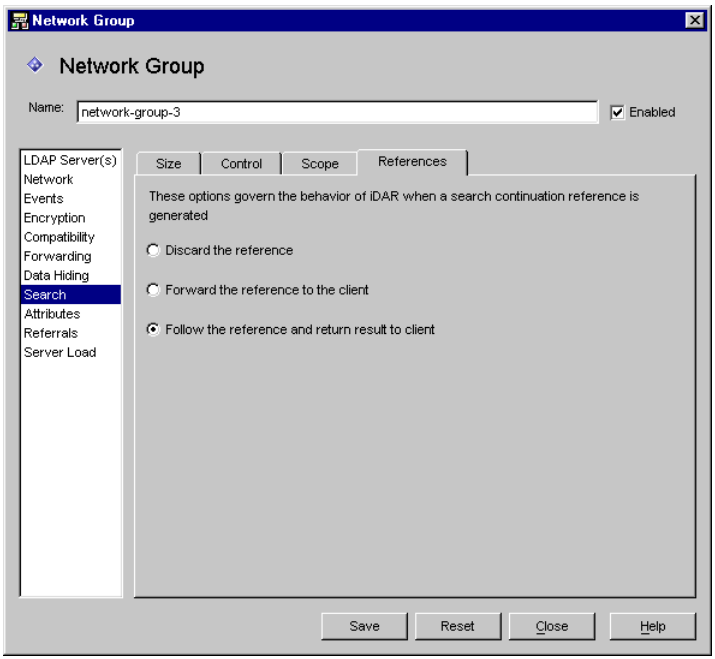
The description of the elements in the Scope tab are as follows:

Permit all search scopes. By default, this option is enabled, permitting all search scopes by a client.

Only ‘base’ search scope is permitted. Enable this option to permit only base search scope.

Only ‘base’ and ‘one level’ searches are permitted. Enable this option to permit only base and one level searches.

18. Select the References tab and specify what to do if a search-continuation reference is generated during a search.



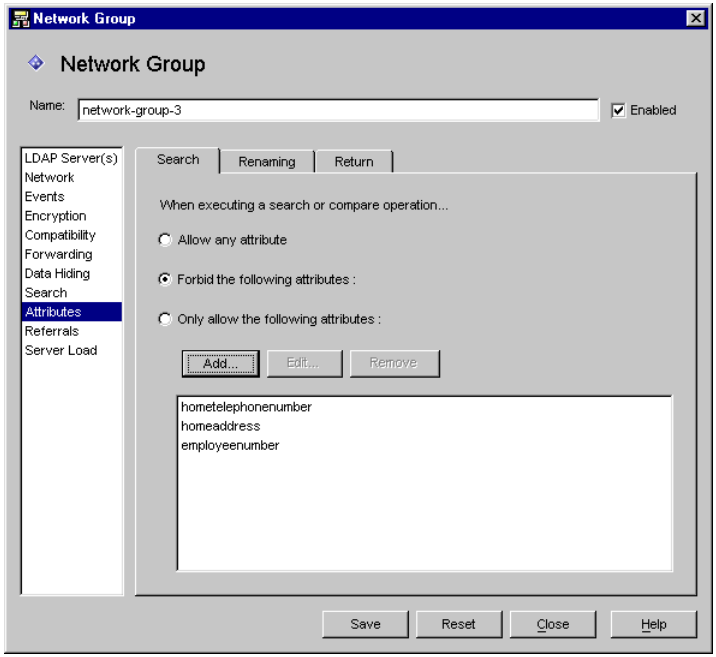
The description of the elements in the References tab is as follows:

Discard the reference. By default, this option is enabled, which will discard a reference if it is generated during a search.

Forward the reference to the client. Enable this option only to forward a search continuation reference.

Follow the reference and return result to client. Enable this option to follow and return the result for a search continuation reference. A search continuation referral is a special case of a referral whereby part of the query has been satisfied by the original directory server queried but that directory server has a reference to another directory server with more data satisfying the query. This option can be used to hide the part of your Directory Information Tree whose naming context is mastered by another LDAP server. It also prevents clients from finding out the network address and port on which this server runs.

19. If you want to specify attribute criteria for the group, select Attributes on the left frame and specify the appropriate values on the right frame.



The description of the elements in the Search tab is as follows:

This tab is used to prevent certain kinds of search and compare operations from reaching the LDAP server. If the client's request falls under this restriction, iDAR will return an insufficient access error to the client.

Allow any attribute. By default, this option is enabled to permit all attributes to be used for search filters and comparisons.

Forbid the following attributes. Enable this option to specify the name of an attribute or attributes that cannot be used by a client in a search filter or compare request.

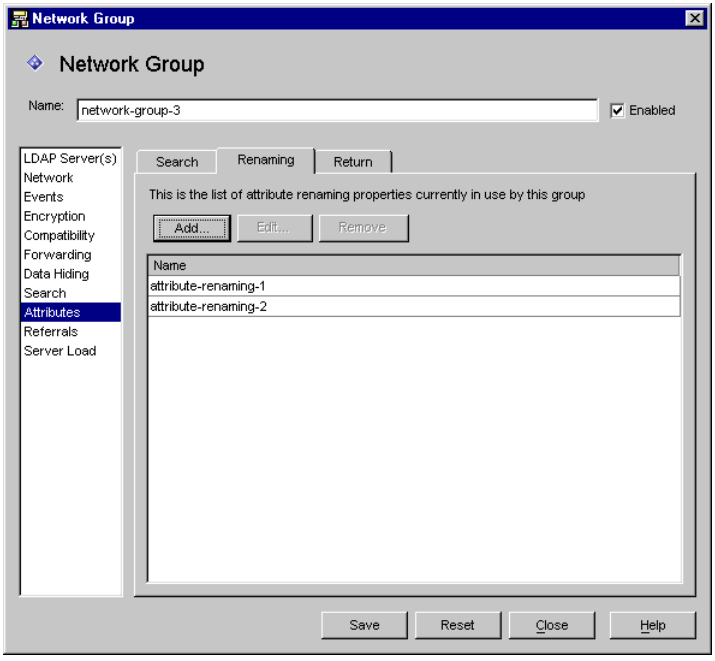
Only allow the following attributes. Enable this option to specify the name of an attribute or attributes that may be used in a search filter or compare request. If there is one or more attributes values present in a network group table and a compare does not match one of these, the request will be rejected by iDAR. If there are no attributes present in a network group table, and an attribute does not match any attributes, then it may be used by clients. For example, if you want only the cn, dn, and mail attributes to be searchable by the client, add these attributes to the table.

Add. Displays a dialog box that allows an attribute to be added to the table. You must specify above whether these attributes are to be forbidden or permitted.

Edit. Displays a dialog box to edit a selected attribute in the table.

Remove. Removes an attribute from the table.

20. Select the Renaming tab and specify the rules for renaming of attributes.



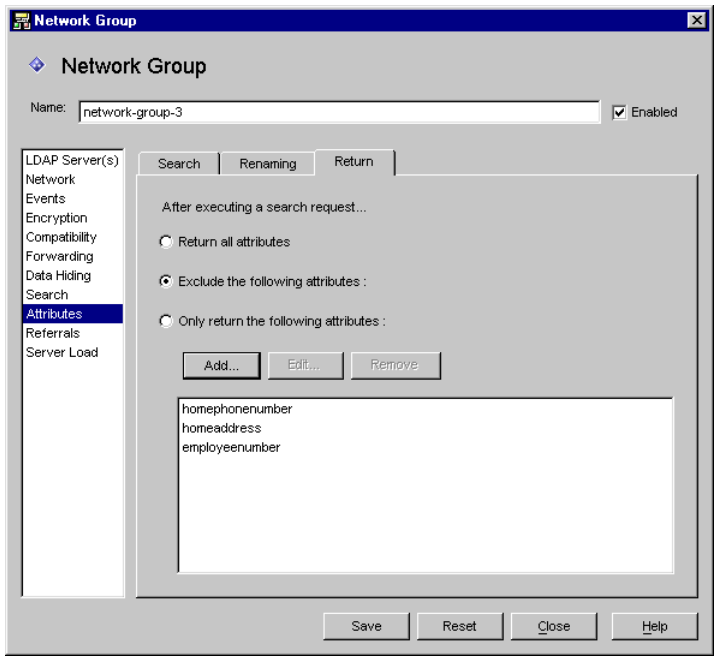
The description of the elements in the Renaming tab is as follows:

Add. Displays a dialog box to add one or more existing attribute renaming properties to the following table that will be used by this network group. (See “Attribute Renaming Property,” on page 114.)

Edit. Displays a dialog box to edit a selected attribute renaming property.

Remove. Remove an attribute renaming property from the table.

21. Select the Return tab and specify restrictions that are to applied to search results being returned by the server, before they are forwarded to the client.



The description of the elements in the Return tab is as follows:

Return all attributes. This option is enabled by default, and it will permit all attributes to be returned.

Exclude the following attributes. Enable this option to specify the name of the attributes that are to be excluded from search result entries.

Only return the following attributes. Enable this option to specify the name of attributes that may be returned from a search result, if present.

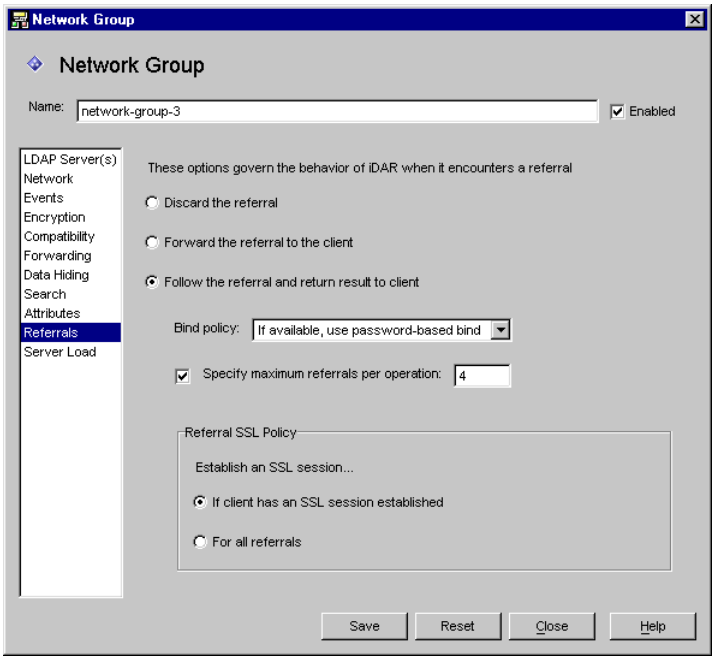
If attributes returned as part of a search result are not present in the “Only return the following attributes” table, they are not returned. If the table is empty and they are not in the “exclude the following attributes” table, they are returned.

Add. Displays a dialog box that allows an attribute to be added to the table. You must specify above whether these attributes are to be forbidden or permitted.

Edit. Displays a dialog box to edit a selected attribute in the table.

Remove. Removes an attribute from the table.

22. If you want to specify referrals for the group (for example, whether the group will forward, follow, or discard referrals returned by the server), select Referrals on the left frame and specify the appropriate values on the right frame.



The description of the on-screen elements is as follows:

Discard the referral. Enable this option if a network group will discard all referrals returned by the server.

Forward the referral. By default, this option is enabled, which will forward referrals returned by the server.

Follow the referral and return result to client. Enable this option if a network group will forward referrals returned by the server and return results to the client.

Bind policy. This option controls the bind policy when an operation is referred and the referral is being followed.

Note that iDAR cannot replay binds for clients bound using a SASL mechanism. Thus the referral operation will be rejected if “Required” is specified and the client used a SASL mechanism to bind.

Always. Select this option if iDAR should always bind anonymous while following referral for a client connected to this network group.

Any. Select this option if a network group should use simple bind if the client had used password-based bind, else bind as anonymous. This is the default.

Required. Select this option if a network group should reject the referred operation if the client is not password-based bound.

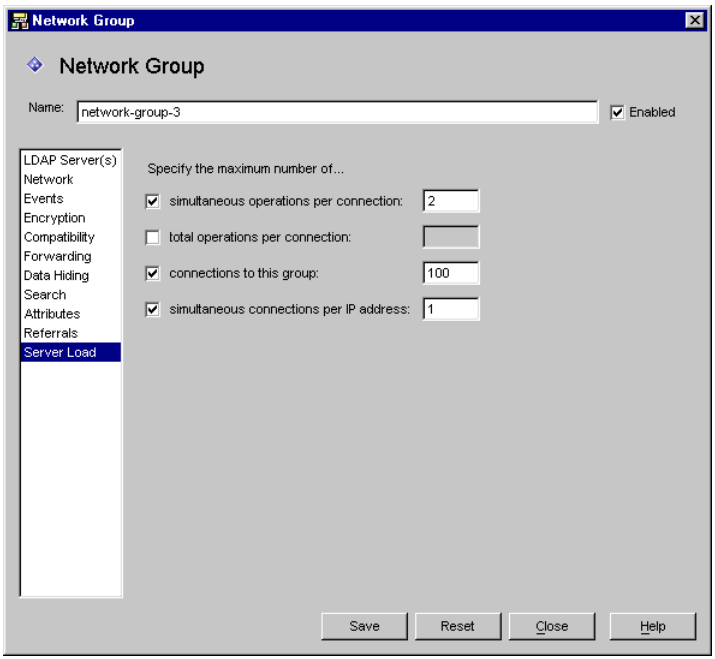
Maximum referrals per operation. Enter an integer value greater or equal to zero. This will limit the maximum number of references that will be followed for a single operation. The default is 15. A value of zero indicates that no limit will be applied.

Referral SSL Policy. In order to enable the Referral SSL Policy Panel, “SSL is available” option must be enabled on the encryption view.

If client has an SSL session established. Enable this option if a network group will only initiate SSL if the client already has SSL session established with iDAR. This is the default.

For all referrals. Enable “For all referrals” if, upon a referral, a group will initiate an SSL session before the operation is forwarded.

23. If you want to specify server load criteria for the group, select Server Load on the left frame and specify the appropriate values on the right frame.



The description of the on-screen elements is as follows:

Simultaneous operations per connection. Select this option to limit the number of simultaneous operations iDAR will process per connection in that group. The value is an integer greater than zero. If this attribute is not present, then no limit is enforced. For example, if you set this value to 1, all the clients in that group will be forced to perform synchronous LDAP operations. Additional simultaneous requests, except for requests to abandon an operation, will fail with Server Busy error.

Total operations per connection. Select this option to limit the total number of operations that iDAR will allow per connection in a group. The value is an integer greater than zero. If a client exceeds the maximum number of operations allowed for its group on one connection, then that connection will be closed by iDAR. If this attribute is not present, then no limit is set.

Connections to this group. Select this option to limit the number of simultaneous connections to this network group, and specify the number.

Simultaneous connections per IP address. Select this option to restrict the number of simultaneous connections clients can make from a single IP address. By default, any number of connections are allowed.

24. Click Save to create the group.

    The iDAR configuration is modified, and you are prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

25. Repeat Step 3 through Step 24 to create any additional groups.

26. Go to the Network Groups window (see Step 2) and prioritize the groups appropriately.

27. Restart the servers; see "Restarting iDAR," on page 57.

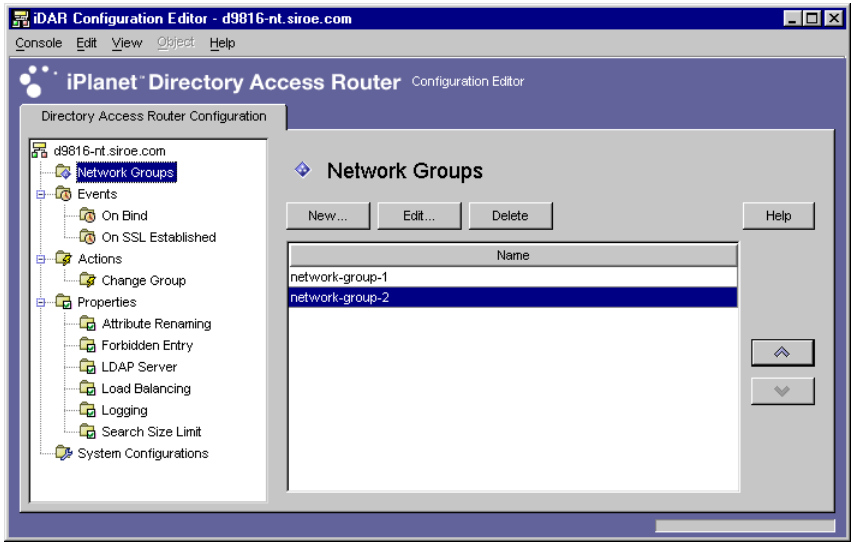
Modifying Groups

To modify a group:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

2. In the navigation tree, select Network Groups.

   The right pane shows the list of existing groups.



3. In the list, select the group you want to modify and click Edit.

4. Make the required modifications.

5. Click Save to save your changes.

   The iDAR configuration is modified, and you are prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

6. Repeat Step 3 through Step 5 to modify any additional groups.

7. Restart the servers; see "Restarting iDAR," on page 57.

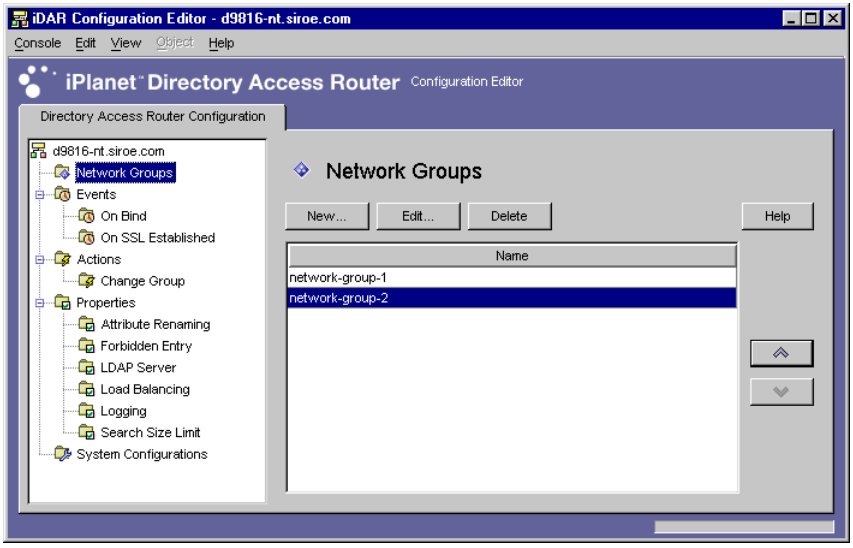
Deleting Groups

You can delete any unwanted network groups from the iDAR configuration. To delete a group:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

2. In the navigation tree, select Network Groups.

   The right pane shows the list of existing groups.



3. In the list, select the group you want to delete and click Delete.

4. Confirm your action.

   The name of the group you deleted is now removed from the list. The iDAR configuration is modified, and you are prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

5. Repeat Step 3 and Step 4 to delete any additional groups.

6. Restart the servers; see "Restarting iDAR," on page 57.

Chapter 7

Defining and Managing Property Objects

As explained in the deployment chapter of this book, iPlanet Directory Access Router (iDAR) can function as an *LDAP access router*, helping you protect your private directory information from unauthorized access, while making it safe for you to publish your public information. The server can handle thousands of LDAP client requests and can apply fine-grained access control rules and protocol filtering rules to each request before routing it to a directory server.

Property objects in iDAR enable you to specify specialized restrictions that the LDAP clients must follow. These properties can then be included in other entries where the restrictions need to be applied. This chapter provides an overview of each of the properties and explains how to create property objects using the iDAR Configuration Editor Console.

The chapter has the following sections:

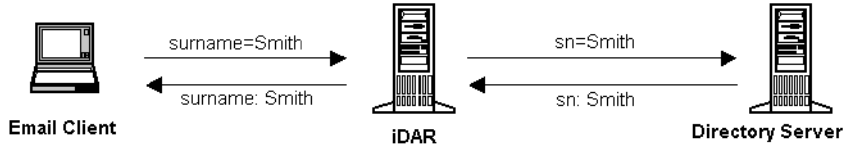
Attribute Renaming Property

Typically, an LDAP directory contains information about entities such as people in your organization and your network resources. For each entity, there would be an entry in the directory. Each entry in a directory is identified by its distinguished name (DN) and is represented by a set of attributes and their values. Each entry has an object class attribute that specifies the kind of object the entry describes and defines the set of additional attributes it contains. Each attribute describes a particular trait or characteristic of an entry. For example, an entry might be of an object class `organizationalPerson`, indicating that the entry represents a person within a particular organization. This object class allows the `givenname` and `telephoneNumber` attributes. The values assigned to these attributes give the name and phone number of the person represented by the entry.

In many directory deployments, the attributes defined on the LDAP client side don't map to the ones defined on the server side. To facilitate communication between the clients and servers in such a setup, iDAR supports renaming of attributes—that is, iDAR can rename attributes in a client query to a form understood by a directory server before passing the query to a directory server, and do the same in the server response before passing it to a client.

Figure 7-1 illustrates how attribute-renaming feature of iDAR can be used for schema mapping.

Figure 7-1 Mapping Schema Using the Attribute Renaming Property



Notice that the email client expects the last names of people to be the value of an attribute named "surname," whereas in the LDAP server, the last names are specified by the attribute named "sn." When iDAR maps these two attributes, only the attribute names are affected; the attribute values remain unchanged.

You use the Attribute Renaming property to define the rules that govern renaming of client and server attributes. You specify the names of the client attributes that need to be mapped to the corresponding server attributes and vice versa. This way, if a client request contains an attribute name unknown to the server, iDAR would be able to map it to a name known to the server and help the client communicate with the server. Similarly, when the server responds back, iDAR would translate any attributes that are unknown to the client to known forms.

The section that follows explains how to create an object for the attribute renaming property from the iDAR Configuration Editor Console.

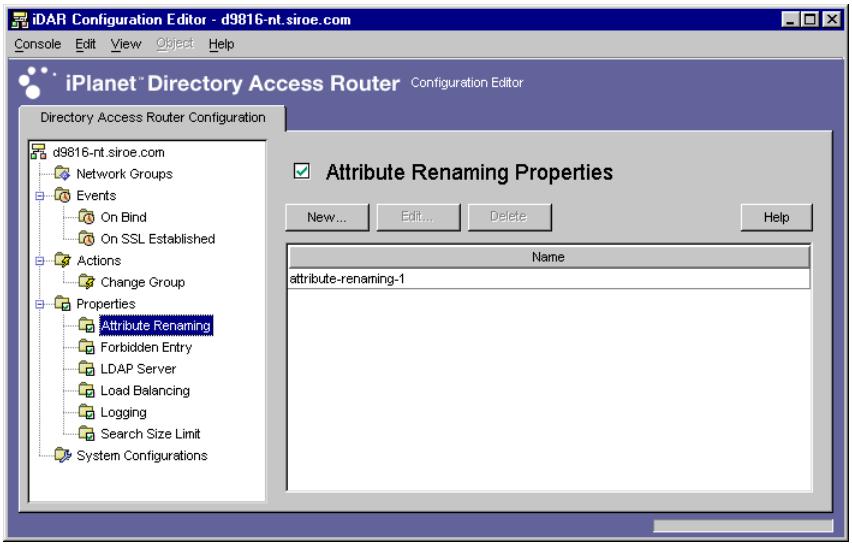
NOTE Any object you create for the attribute-renaming property must have both server and client attributes. Otherwise, iDAR will fail to start.

Creating Attribute Renaming Property Objects

To identify the client and server attributes that iDAR should rename:

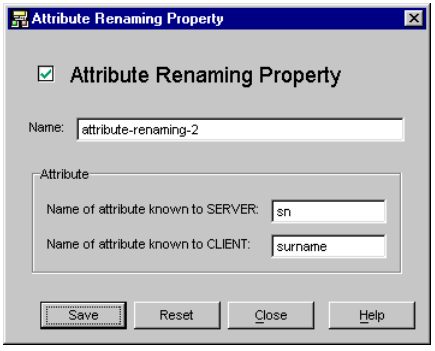
1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, expand the Properties node, and then select Attribute Renaming.

   The right pane shows the list of existing objects for the attribute renaming property.



3. Click New.

   The Attribute Renaming Property window appears.



4. In the Name field type, a name for the property object. The name must be a unique alphanumeric string.

5. In the remaining fields, identify the attributes for mapping:

   Name of attribute known to SERVER. Enter a value to specify the name of the attribute known to the server.

   Name of attribute known to CLIENT. Enter a value to specify the name of the attribute known to the client.

If a client request contains an attribute name specified by the "Name of attribute known to CLIENT," it will be transformed to the value of "Name of attribute known to SERVER." Similarly, if a result sent by the server contains an attribute name specified in "Name of attribute known to SERVER," it will be transformed to the value of "Name of attribute known to CLIENT."

6. Click Save to create the object.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

7. Repeat Step 3 through Step 6 to create any additional objects.

8. Restart the servers; see "Restarting iDAR," on page 57.

Forbidden Entry Property

For various reasons, certain entries (or the attributes that represent these entries) in an LDAP directory will need to be hidden from the LDAP clients. For example, if your directory contains entries for all the employees and each of these entries contain relevant attributes for employee data, such as the name, email address, department, office location, office phone number, and home phone number, you can hide all employees' home phone numbers from being visible to clients.

A *forbidden entry* refers to an entry in an LDAP directory that needs to be hidden from LDAP clients. To facilitate communication between the clients and directory servers in such a setup, iDAR supports forbidden entries—that is, iDAR can hide LDAP entries and the attributes of these entries from LDAP clients.

You use the Forbidden Entry property to define the rules that govern hiding of directory entries and their attributes. This property enables you to specify a list of entries or the attributes of the entries that need to be hidden in several ways. For example, you can specify:

- DNs of entries or attributes in those entries that you want to hide.
- Regular expressions of DNs of entries or attributes in those entries that you want to hide (for example, `.*OU=INTERNAL.*`).
- Attribute name/value pairs of an entry (for example, `secret:yes`). If an entry has an attribute name/value pair that matches any of the specified attribute name/value pairs, then that entry or some of its content will be hidden.

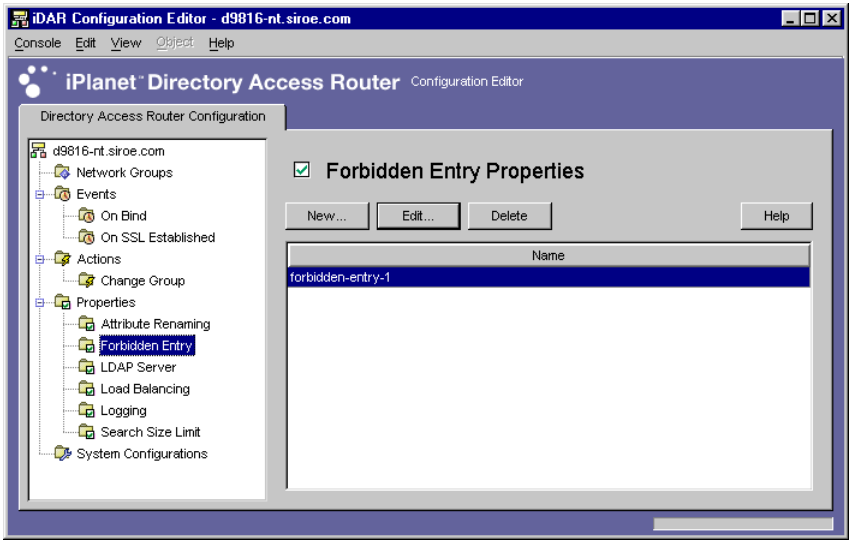
The section that follows explains how to create an object for the forbidden entry property from the iDAR Configuration Editor Console.

Creating Forbidden Entry Property Objects

To identify the entries or attributes of any entries that iDAR should hide from clients:

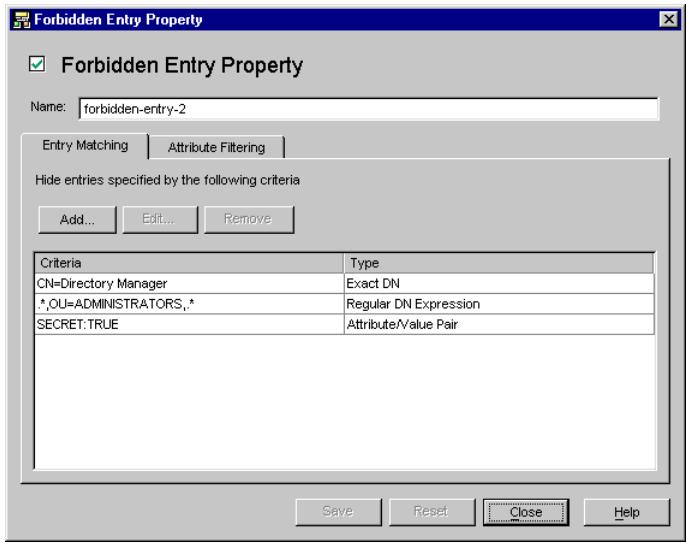
1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, expand the Properties node, and then select Forbidden Entry.

   The right pane shows the list of existing objects for the forbidden entry property.



3. Click New.

   The Forbidden Entry Property window appears.



4. In the Name field type a name for the property object. The name must be a unique alphanumeric string.

5. In the Entry Matching tab, specify the appropriate values; the tab displays settings for this property's name and LDAP entries to hide.

   Add. Displays a menu for adding criteria for hiding LDAP entries. Criteria can be of the following type: Exact DN, Regular DN Expression, or Attribute / Value Pair.

   Exact DN. Displays a dialog for entering the DN of an entry to hide.

   Regular DN expression. Displays a dialog for entering a regular DN expression of entries to hide. The regular expression of the DN should be specified in the normalized form; that is, there should be no spaces between RDN components and the "=" sign and attribute names and values must be in all capital letters.

   For example, to match any DN with a RDN component of "ou=internal," you must specify the following:

   .*OU=INTERNAL.*

   If the Attribute Filtering tab contains attribute names to be included, and an attribute does not match one of those listed, then it is not returned. If an LDAP entry has no attributes that match any attributes to be excluded in the Attribute Filtering tab, then it is returned.

The following book can be used as a reference on regular expressions: *Mastering Regular Expressions*, by Friedl and Oram, published by O'Reilly, ISBN: 1565922573.

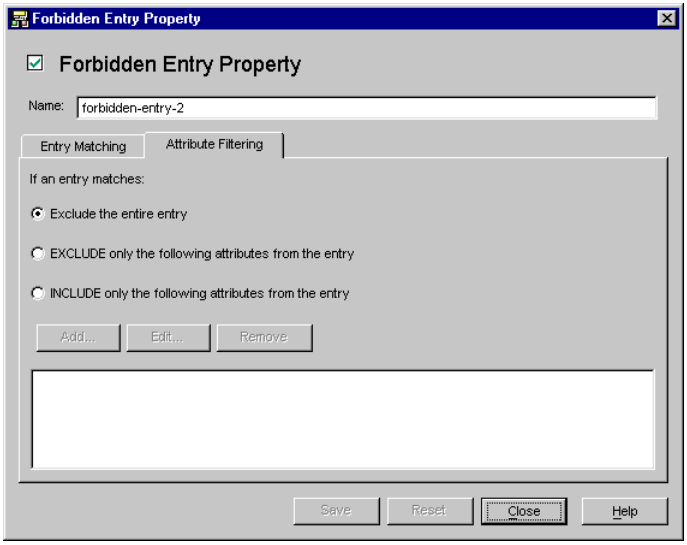
Attribute/Value pair. Displays a dialog used to specify an attribute name/value pairs. If an entry has an attribute name/value pair that matches any of the specified attribute name/value pairs, then that entry or some of its content will be hidden.

For example, if you want to restrict all entries that have either “ou=internal” or “secret=yes” as one if its attributes, then you can specify the following: an attribute of “ou” and a value of “internal.”

Edit. Displays a dialog for editing the currently-selected entry in the table.

Remove. Removes the currently-selected entry in the table.

6. Select the Attribute Filtering tab, and specify the appropriate values.



The tab contains settings that allow certain attributes to be excluded, or specifically included:

Exclude the entire entry. Select this option to indicate that no attribute filtering is to be performed and that the entire entry is to be hidden.

EXCLUDE only the following attributes from the entry. Select this option to indicate that the table contains a list of attribute names that are to be excluded from the entry that has matched any of the above specifications.

INCLUDE only the following attributes from the entry. Select this option to indicate that the table contains a list of attribute names that may be returned as part of the entry that has matched any of the above specifications.

7. Click Save to create the object.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

8. Repeat Step 3 through Step 7 to create any additional objects.

9. Restart the servers; see "Restarting iDAR," on page 57.

LDAP Server Property

In a directory deployment, iDAR is located between LDAP clients and LDAP directory servers. It filters requests from LDAP clients before routing them to LDAP directory servers and responses from directory servers before passing them to the clients. iDAR also supports automatic load balancing and automatic failover and failback among a set of replicated directory servers.

You use the LDAP Server property to identify the directory servers that iDAR should use as the backend servers. When defining this property, you specify all the details required by iDAR—for example, the IP address or fully-qualified hostname of the directory server, the port number at which the directory server is listening for client connections, the LDAP versions supported by the server, the version to be used for communication between iDAR and this server, and so on—to communicate with a directory server.

The section that follows explains how to create an object for the LDAP server property from the iDAR Configuration Editor Console.

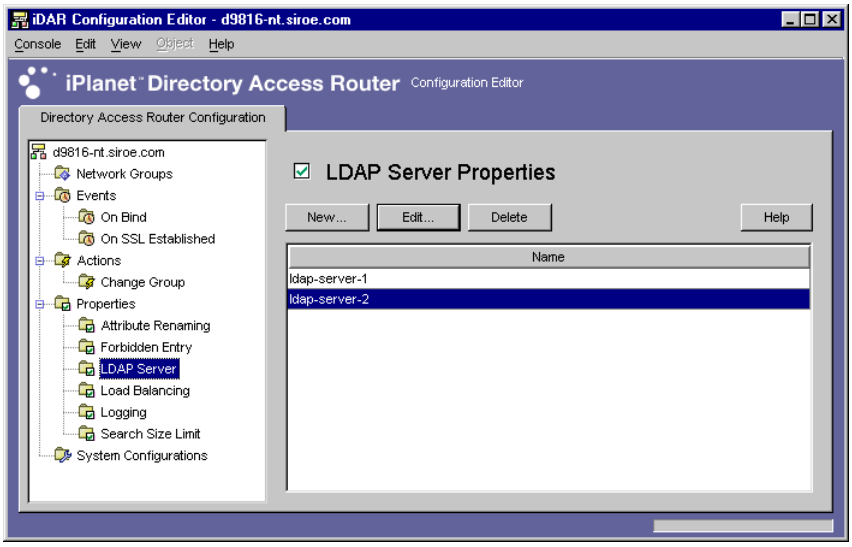
Creating LDAP Server Property Objects

To identify the directory servers that iDAR should communicate with:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

2. In the navigation tree, expand the Properties node, and then select LDAP Server.

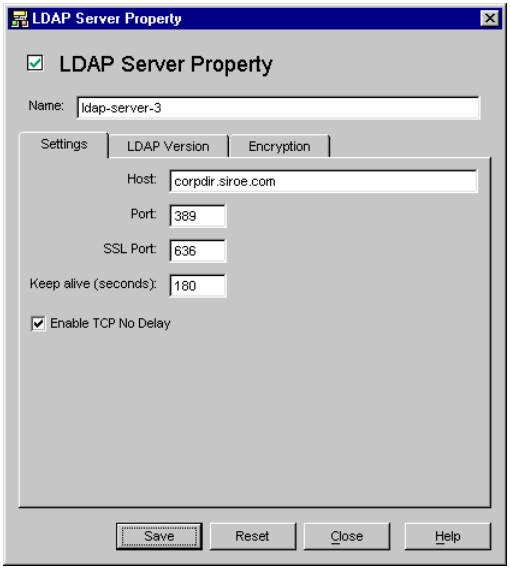
   The right pane shows the list of existing objects for the LDAP Server property.



3. Click New.

   The LDAP Server Property window appears.

4. In the Name field, type a name for the property object. The name must be a unique alphanumeric string.



5. In the Settings tab specify the basic settings of the LDAP server referred to by this property.

   Host. Enter a value specifying the full domain name or IP address of the host where the backend LDAP server is running. This attribute is mandatory.

   Port. Enter a number specifying the port on which the backend LDAP server is running. The default port used, if this attribute is absent, is 389.

   SSL port. Enter a number specifying the port on which the backend LDAP server listens for LDAPS (LDAP over SSL) connections. Do not set any value for this attribute if the backend LDAP server does not support LDAPS.

   Keep alive interval. Enter the number of seconds after which iDAR will poke an unresponsive server, to determine if the network link to an LDAP directory server is down or if the LDAP directory server has become unresponsive. If the client connected to iDAR has pending operations and if iDAR has not received any data from the connection's LDAP server for the number of seconds specified here, then iDAR will test the availability of the LDAP server by opening another communication channel to it. If iDAR is unsuccessful in doing so, it will fail over to another LDAP server, if available. The default value for this attribute is 180 seconds. It is recommended that you increase this value if the LDAP server is not on the same local network as iDAR.

   Enable TCP no delay. Disable this option to cause iDAR to use the Nagel Algorithm on connections to this server. The option must be disabled only if the network bandwidth between iDAR and the server defined by this object entry is very limited. By default, this setting is enabled.

6. Select the LDAP Version tab and specify the appropriate values.

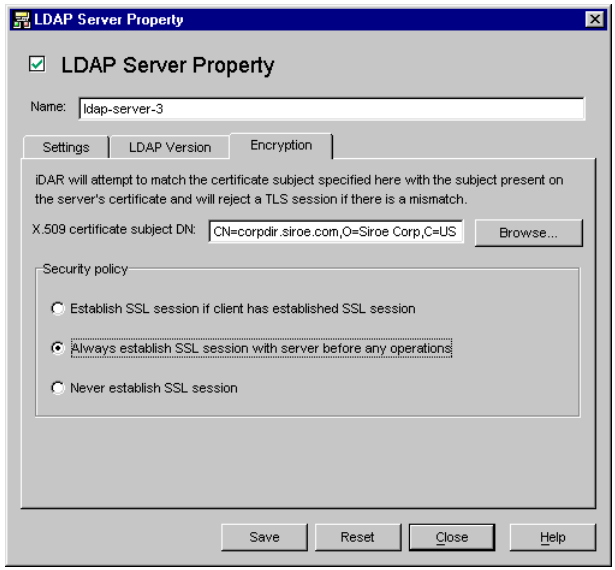
The tab displays settings indicating which versions of LDAP are supported by this server, and which version should be used for communication between iDAR and this server.

Supported LDAP version. Select one of the two options present: LDAP versions 2 and 3, or LDAP version 2 only. The default is LDAP versions 2 and 3.

LDAP version to use. Select one of the three options present: “Whichever version the client is using,” “LDAP version 3 only,” or “LDAP version 2 only.” This attribute tells iDAR the preferred LDAP protocol version to use when talking to the backend server this entry defines. By default, “Whichever version the client is using” is selected.

This option is useful when you have an LDAPv2 client for which iDAR needs to follow referrals. In this case iDAR itself needs to connect as LDAPv3 client to the backend server in order for the backend server to send referrals back to it. LDAP version 3 only must be selected if the network group referring to this property allows multiple LDAP version 2 binds.

7. Select the Encryption tab and specify the appropriate values.



The tab displays settings relating to secure communications for the LDAP server referred to by this property.

X.509 certificate subject DN. Specify the LDAP server's certificate subject name. If specified, iDAR will attempt to match the certificate subject with the subject present on the LDAP server's certificate and will reject a TLS session if there is a mismatch. (This attribute allows iDAR to authenticate the LDAP server to which it is connecting. iDAR accepts any name if this attribute is not set.)

Security policy. Select one of the options that define the security policy for connections between iDAR and the backend server: "Establish SSL session if client has established SSL session," "Always establish SSL session with server before any operations," "or Never establish SSL session."

8. Click Save to create the object.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

9. Repeat Step 3 through Step 8 to create any additional objects.

10. Restart the servers; see "Restarting iDAR," on page 57.

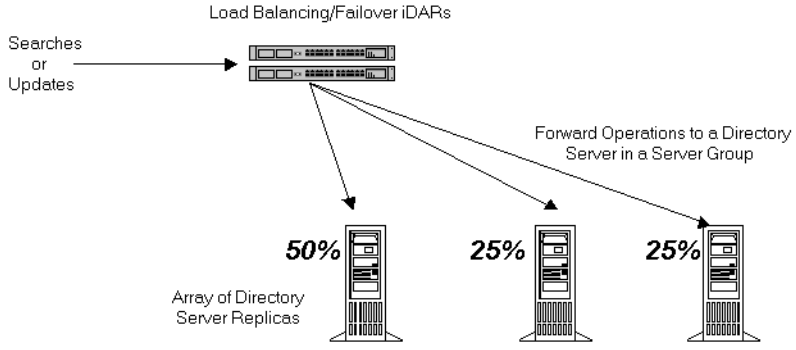
Load Balancing Property

iDAR enables high availability of directory deployments by providing both automatic load balancing and automatic failover and failback among a set of replicated LDAP directory servers. In order for iDAR to do this, you need to identify the directory servers that iDAR should work with and specify how client load is to be distributed among these servers.

You configure iDAR for load balancing using the Load Balancing property. This property enables you to identify the back-end directory servers that iDAR should communicate with and specify the percentage of total client load each directory server should receive. Once configured, iDAR automatically distributes client queries to different directory servers conforming to the load criteria defined in the configuration. If a directory server becomes unavailable, iDAR distributes the load percentage of that server proportionally among the available servers based on their load percentage. iDAR starts rejecting client queries if all back-end LDAP servers become unavailable.

Figure 7-2 shows client load distributed among a set of three directory server replicas.

Figure 7-2 Load Balancing Across a Set of LDAP Directory Replicas



Load balancing in iDAR is session based. This means that the decision function that chooses a particular directory server to which a client's queries will be directed is applied once per client session, in particular, at the start of the client session. All subsequent client queries in that session are directed to the same directory server that was chosen at the beginning of the session.

The number of back-end directory servers that iDAR can load balance depends on several factors, some of which are listed below:

- The size of the host running iDAR
- The network bandwidth available
- The query mix that iDAR receives
- The length of client sessions
- The iDAR configuration

In general, iDAR can support fewer directory servers if most sessions are short lived and queries are computationally intensive. Computationally intensive queries are those that require the inspection of the entire message such as, if the attribute renaming (see “Attribute Renaming Property,” on page 114) feature is used.

iDAR detects when a directory server becomes unavailable either when a connection attempt is returned with a connection refused error or when it times out. Because both these cases occur at the initial stages of the session, and no operations have yet been processed for that session, iDAR fails over to another server provided one is available transparently. In the connect-attempt-timeout case, the client can experience significant delay in getting a response. If a connection between iDAR and a back-end server is abruptly lost, iDAR returns `LDAP_BUSY` error for all outstanding operations to the affected client. Subsequently, iDAR fails over that client session to another directory server.

In order to avoid iDAR from becoming the single point of failure for your directory deployment, we recommend you use at least two iDARs with an IP appliance in front of it. This is described in Chapter 2, “iDAR Deployment Scenarios.” In case it is not possible to deploy iDAR this way, we recommend that you use the -M switch (see “Supported Flags,” on page 198), which will enable iDAR to monitor itself.

iDAR uses a monitor process to make health checks on its backend servers. This feature is automatically enabled if load balancing is used. iDAR makes an anonymous search operation for the Root DSE every 10 seconds for each of its backend directory servers. If one of them becomes unavailable or unresponsive, iDAR removes it from the active load balanced server set. When the server becomes available again, it is reintroduced in the set. In order for the monitoring feature to work efficiently, you must have configured the host on which iDAR is running according to the recommendations of the `<code>idsktune</code>` utility described in Chapter 2, “Computer System Requirements” of the *iDAR Installation Guide*.

The section that follows explains how to create an object for the load balancing property from the iDAR Configuration Editor Console.

NOTE Any object you create for the load balancing property must have at least one LDAP Server property and the percentages must add up to 100 percent. Otherwise, iDAR will fail to start.

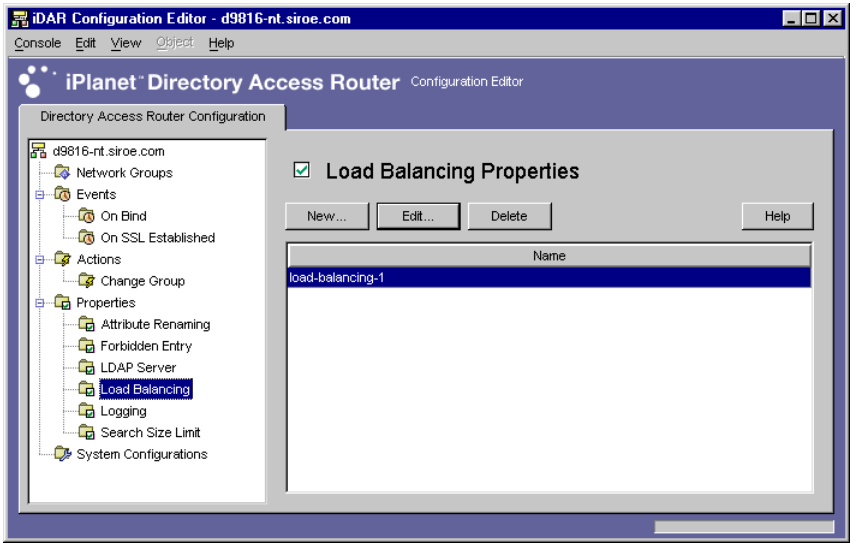
Creating Load Balancing Property Objects

This section explains how to configure iDAR for load balancing. Before you create objects for the load balancing property, be sure to identify the LDAP directory servers that iDAR should use for balancing the client load. For details, see “LDAP Server Property,” on page 121.

To define how iDAR should balance load among a set of directory servers:

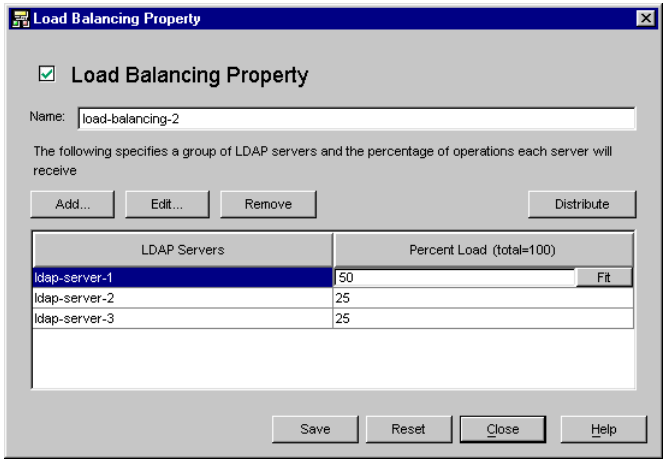
1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, expand the Properties node, and then select Load Balancing.

   The right pane shows the list of existing objects for the Load Balancing property.



3. Click New.

   The Load Balancing Property window appears.



4. In the Name field, type a name for the property object. The name must be a unique alphanumeric string.

5. Use the remaining form elements to get the desired results.

   To edit a percentage, click the Percentage Load column next to the row containing an LDAP Server, type a number between 0 and 100, and click the Fit button. This action assigns the percentage to the current row and attempts to make the sum of all the percentages 100. The current percentage sum is displayed in the Percentage Load column heading.

   Enabled. By default, this option is selected for you. Deselect it to disable a property in a configuration. For a property to be part of iDAR configuration, this option must be selected.

   Add. Displays a dialog for adding a reference to an LDAP server property. By default, the first server added is assigned 100 percent of the load with subsequent additions getting 0 percent.

   Edit. Displays a dialog for editing the currently-selected item from the table.

   Remove. Removes the currently-selected LDAP server from the list of servers across which load balancing will be performed.

   Distribute. Distributes the percentage load evenly across all LDAP servers currently referred to in the table.

6. Click Save to create the object.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

7. Repeat Step 3 through Step 6 to create any additional objects.

8. Restart the servers; see "Restarting iDAR," on page 57.

Logging Property

You use the Logging property to define the logging characteristics of iDAR. Chapter 10, "Configuring and Monitoring Logs" provides an overview about this property and explains how to use the property for configuring iDAR logs.

Search Size Limit Property

An LDAP directory typically functions as a central repository for an organization, enabling LDAP clients deployed across the organization to look up information. LDAP client generally look up information by searching for specific information using search filters. When searching for an entry, clients generally specify attributes associated with that type of entry; for example, when you search for people entries, you can use the `CN` attribute to search for people with a specific common name.

iDAR can handle thousands of LDAP client requests and can be configured to apply fine-grained access control policy on LDAP directories, such as controlling who can perform different types of operations on different parts of the Directory Information Tree (DIT). You can also configure iDAR to disallow certain kinds of operations, such as the ones performed by web trawlers and robots to collect information contained in a directory.

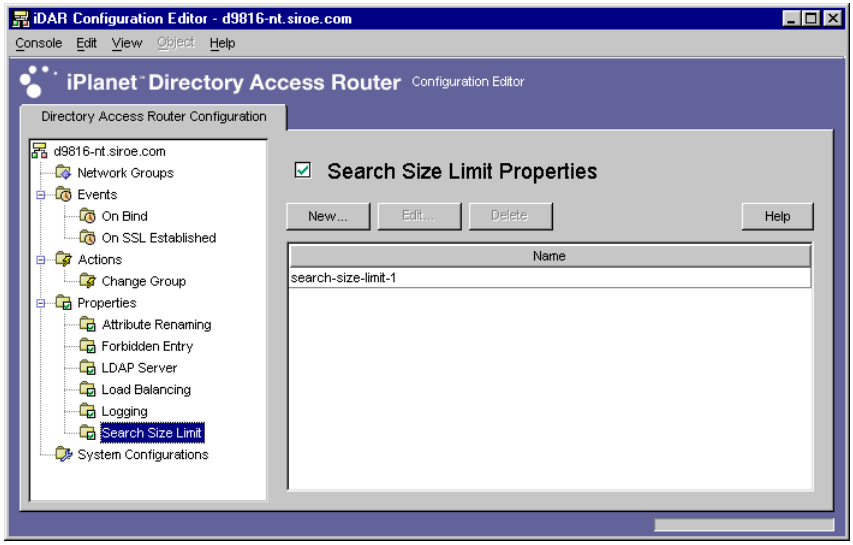
You use the Search Size Limit property to apply size limits based on the search base and search scope. If neither the search base nor search scope specified in this property object entry match a given search, the size limit defaults to the size limit specified in the Network Group object entry; see Chapter 6, "Creating and Managing Groups."

The section that follows explains how to create an object for the search size limit property from the iDAR Configuration Editor Console.

Creating Search Size Limit Property Objects

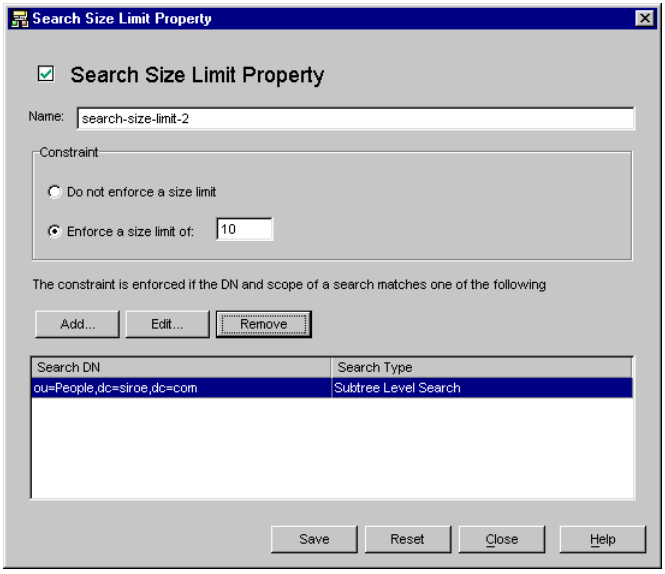
To define how iDAR should limit search sizes:

1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, expand the Properties node, and then select Search Size Limit.



3. Click New.

   The Search Size Limit Properties window appears.



4. In the Name field, type a name for the property object. The name must be a unique alphanumeric string.

5. Use the remaining form elements to get the desired results:

   Constraint. Specifies whether or not to enforce a size limit constraint.

   Do not enforce a size limit. Select this option to specify that no size limit will be enforced.

   Enforce a size limit of. Select this option and enter an integer value, specifying the size limit to enforce.

   Add. Displays a menu for adding a size limit condition. Conditions must be one of two types: one level search and subtree level search.

   One level search. Displays a dialog for entering a DN and adding it to the condition table. If the DN of the search base of a one level search matches one of the distinguished names specified for one level searches from the condition table, the size limit specified is enforced as the size limit of that search.

   Subtree level search. Displays a dialog for entering a DN. If the DN of the search base of a subtree search matches one of the distinguished names specified for subtree level searches from the condition table, the size limit specified is enforced as the size limit of that search.

   Edit. Displays a dialog for editing the currently-selected entry in the table.

Remove. Removes the currently-selected entry in the table.

6. Click Save to create the object.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

7. Repeat Step 3 through Step 6 to create any additional objects.

8. Restart the servers; see "Restarting iDAR," on page 57.

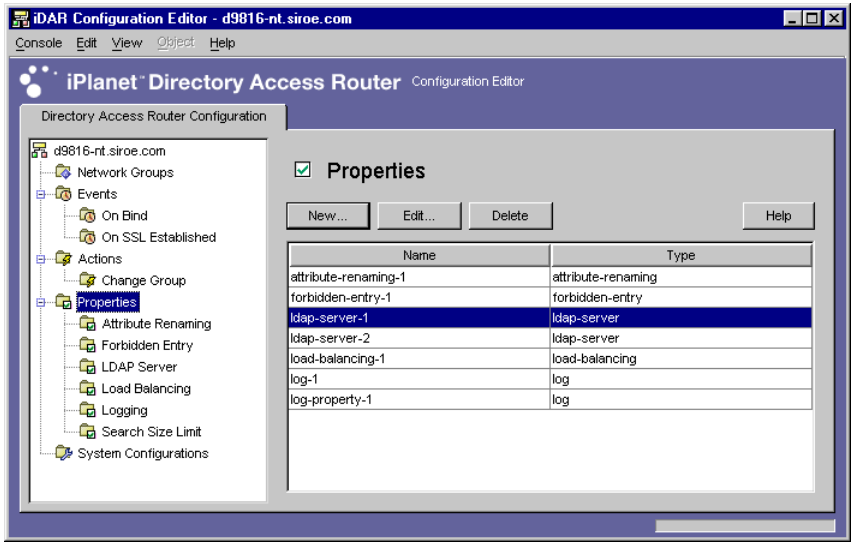
Modifying Property Objects

To modify a property object:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

2. In the navigation tree, select the Properties node.

   The right pane shows the list of existing property objects. To view objects pertaining to a specific property, expand the Properties node, and then select the property of your interest.



3. In the list, select the object you want to modify and click Edit.

4. Make the required modifications.
5. Click Save to save your changes.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.
6. Repeat Step 3 through Step 5 to modify any additional objects.
7. Restart the servers; see "Restarting iDAR," on page 57.

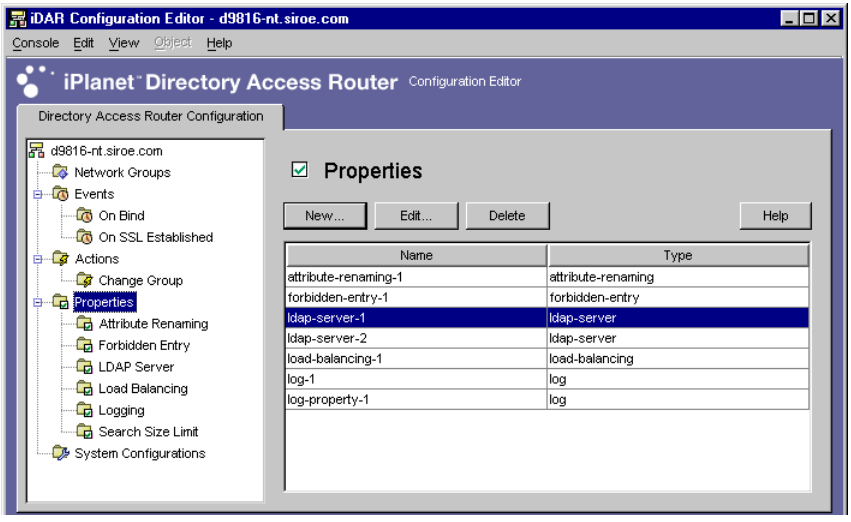
Deleting Property Objects

You can delete any unwanted property objects from the iDAR configuration. Before deleting an object, make sure that it's not used in any other configuration entries.

To delete a property object:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.
2. In the navigation tree, select the Properties node.

   The right pane shows the list of existing property objects. To view objects pertaining to a specific property, expand the Properties node, and then select the property of your interest.



3. In the list, select the object you want to delete and click Delete.
4. Confirm your action.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.
5. Repeat Step 3 and Step 4 to delete any additional objects.
6. Restart the servers; see "Restarting iDAR," on page 57.

Chapter 8

Creating and Managing Event Objects

iPlanet Directory Access Router (iDAR) supports event-driven actions; that is, you can configure iDAR to execute specified actions when specific events occur. This chapter explains how to create and manage event objects using the iDAR Configuration Editor Console.

The chapter has the following sections:

Overview of Events

An *event* is a certain iDAR state at a certain point while it's running. You use event objects to specify conditions that iDAR should evaluate at predetermined states. As a part of defining an event object, you also specify the action iDAR should take if the conditions are satisfied. For details about actions, see Chapter 9, "Creating and Managing Action Objects."

Currently, iDAR can recognize or keep track of two types of events:

- OnBindSuccess event—This event is evaluated when a client successfully completes a bind operation.
- OnSSLEstablished event—This event is evaluated when a client successfully establishes an SSL session. This event does not have any associated conditions and always executes its list of actions.

You can define event objects based on these two events only. For example, you can define an event object for detecting when a client completes a successful bind. A part of this definition could be to take certain action when the event occurs, for example, to change the access group of that client. For details about groups, see Chapter 6, “Creating and Managing Groups.”

Creating Event Objects

This section explains how to create event objects that are based on OnBindSuccess and OnSSLEstablished events. For details about these events, see “Overview of Events,” on page 137.

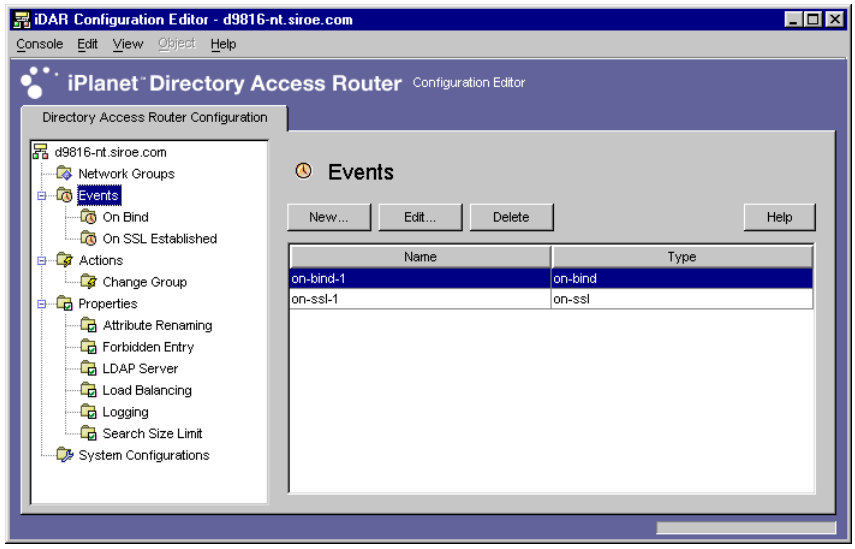
- Creating OnBindSuccess Event Objects
- Creating OnSSLEstablished Event Objects

Creating OnBindSuccess Event Objects

To create an event object based on the OnBindSuccess event:

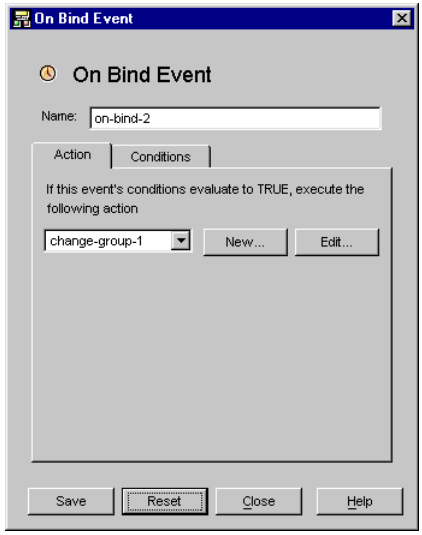
1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, expand the Events node, and then select On Bind.

   The right pane shows the list of existing event objects that are based on the OnBindSuccess event.



3. Click New.

   The On Bind Event window appears.

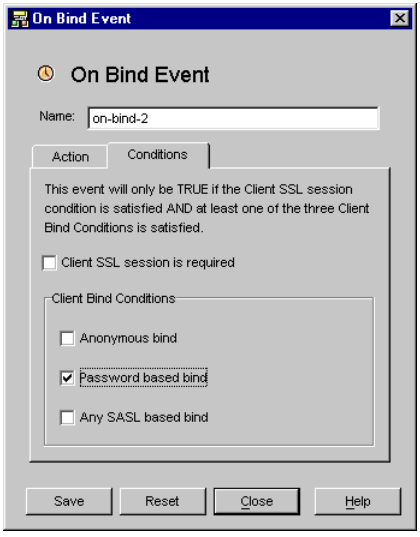


4. In the Name field, type a name for the event object. The name must be a unique alphanumeric string.

5. In the Action tab, select the action to be performed when the event occurs (that is, when the event evaluates to true).

   Click the Edit button to modify the parameters pertaining to the currently-selected action object. You can also define a new action object by clicking the New button.

6. Select the Conditions tab and specify the conditions.



The event will evaluate to TRUE only when the specified conditions are met—that is, the criteria specified in this tab must evaluate to TRUE in order for the action specified in the Action tab to be executed. The conditions will only be TRUE if the Client SSL session condition is satisfied and at least one of the three client bind conditions is satisfied.

Client SSL session is required. Select this option to indicate that the condition will evaluate to TRUE only if the client has established an SSL session with iDAR. The default is FALSE.

Client bind conditions. Conditions are one of the following: “Anonymous bind,” “Password based bind,” and “Any SASL based bind.”

Anonymous bind. Select this option to indicate that the condition will evaluate to TRUE only if the Client SSL session requirement is met and the client has just completed a successful anonymous bind.

Password based bind. Select this option to indicate that the condition will evaluate to TRUE only if the Client SSL session requirement is met and the client has just completed a successful password based bind.

Any SASL based bind. Select this option to indicate that the condition will evaluate to TRUE only if the Client SSL session requirement is met and the client has just completed a successful bind using any SASL mechanism.

7. Click Save to create the event object.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

8. Repeat Step 3 through Step 7 to create any additional objects.

9. Restart the servers; see "Restarting iDAR," on page 57.

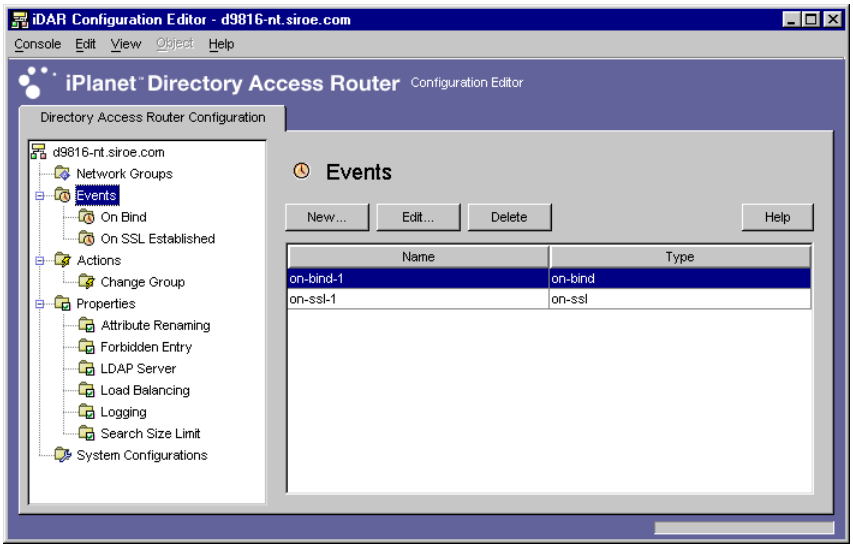
Creating OnSSLEstablished Event Objects

To create an event object based on the OnSSLEstablished event:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

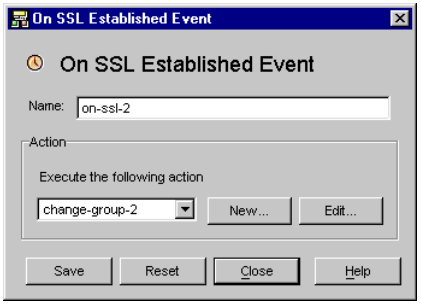
2. In the navigation tree, expand the Events node, and then select On SSL Established.

   The right pane shows the list of existing event objects that are based on the OnSSLEstablished event.



3. Click New.

   The On SSL Established Event window appears.



4. In the Name field, type a name for the event object. The name must be a unique alphanumeric string.

5. In the Action section, select the action to be performed when the event occurs (that is, when the event evaluates to TRUE).

   Click the Edit button to modify the parameters pertaining to the currently-selected action. You can also define a new action by clicking the New button.

6. Click Save to create the event object.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

7. Repeat Step 3 through Step 6 to create any additional objects.

8. Restart the servers; see "Restarting iDAR," on page 57.

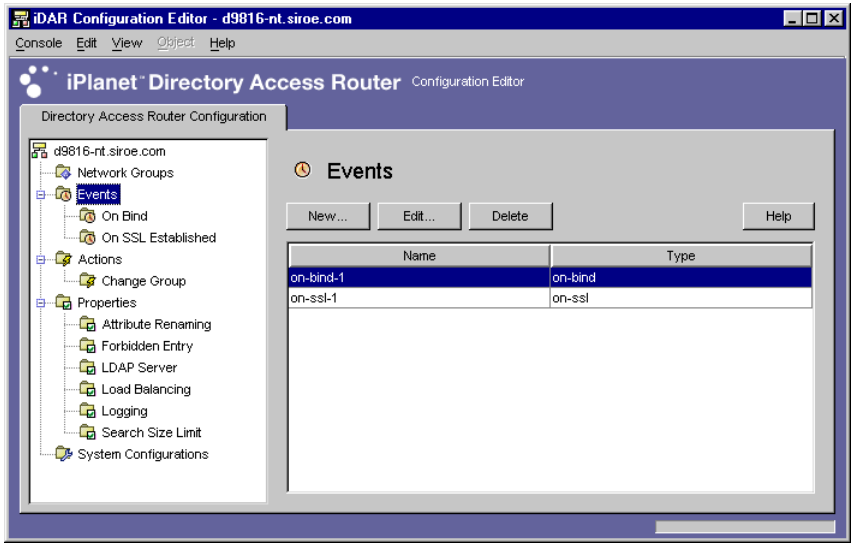
Modifying Event Objects

To modify an event object:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

2. In the navigation tree, select Events.

   The right pane shows the list of existing event objects. To view objects pertaining to an event type, expand the Events node, and then select the event type of your interest.



3. In the list, select the event object you want to modify and click Edit.

4. Make the required modifications.

5. Click Save to save your changes.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

6. Repeat Step 3 through Step 5 to modify any additional objects.

7. Restart the servers; see "Restarting iDAR," on page 57.

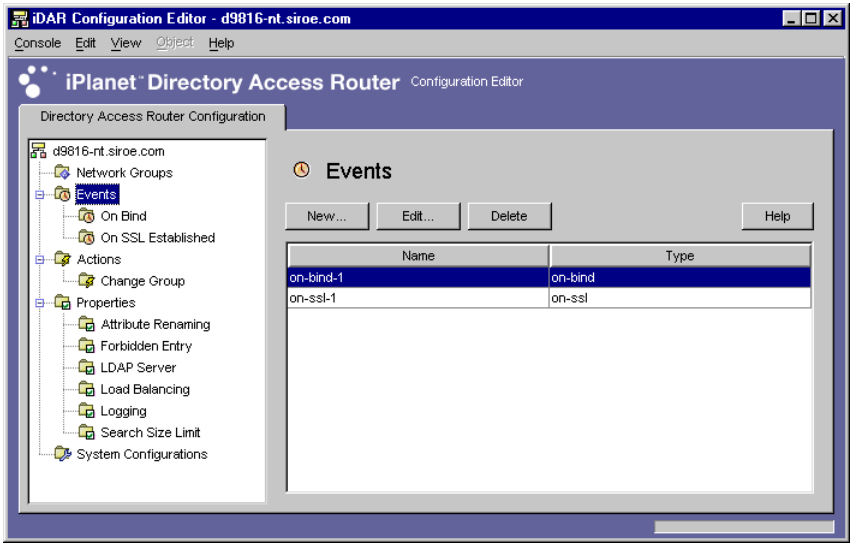
Deleting Event Objects

You can delete any unwanted event objects from the iDAR configuration. To delete an event object:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

2. In the navigation tree, select the Events node.

   The right pane shows the list of existing event objects. To view objects pertaining to an event type, expand the Events node, and then select the event type of your interest.



3. In the list, select the event object you want to delete and click Delete.

4. When prompted, confirm your action.

   The name of the event object you deleted is now removed from the list. The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

5. Repeat Step 3 and Step 4 to delete any additional objects.

6. Restart the servers; see "Restarting iDAR," on page 57.

Chapter 9

Creating and Managing Action Objects

iPlanet Directory Access Router (iDAR) supports event-driven actions, that is, you can configure iDAR to execute specified actions when specific events occur. This chapter explains how to create and manage action objects using the iDAR Configuration Editor Console.

The chapter has the following sections:

Overview of Actions

An *action* refers to a task that iDAR can execute. You use an action object to specify the action iDAR should take if the rules or conditions defined by an event object evaluates to `TRUE`. Event objects are used to specify conditions which are evaluated by iDAR at predetermined states. For details about events, see Chapter 8, “Creating and Managing Event Objects.”

Currently, iDAR can execute one action called `ChangeGroup`. This action enables you to configure iDAR to change a client from one access group to another based on the evaluation of a rule. For details about groups, see Chapter 6, “Creating and Managing Groups.”

The change-group feature is especially useful if your LDAP directory contains information about mobile users, for example, users who connect to the directory from different IP addresses or physical locations. You can setup iDAR in such a way that a mobile user would connect to iDAR with a dynamic IP address and drop into a “default” access group. The “default” access group would have a rule based on the `OnBindSuccess` event, that evaluates to TRUE only if the bind credentials provided by the mobile user are authenticated. This rule would also have the `ChangeGroup` action configured to change the mobile user’s access group from “default” to the access group the mobile user is usually assigned to when accessing iDAR with a static IP address.

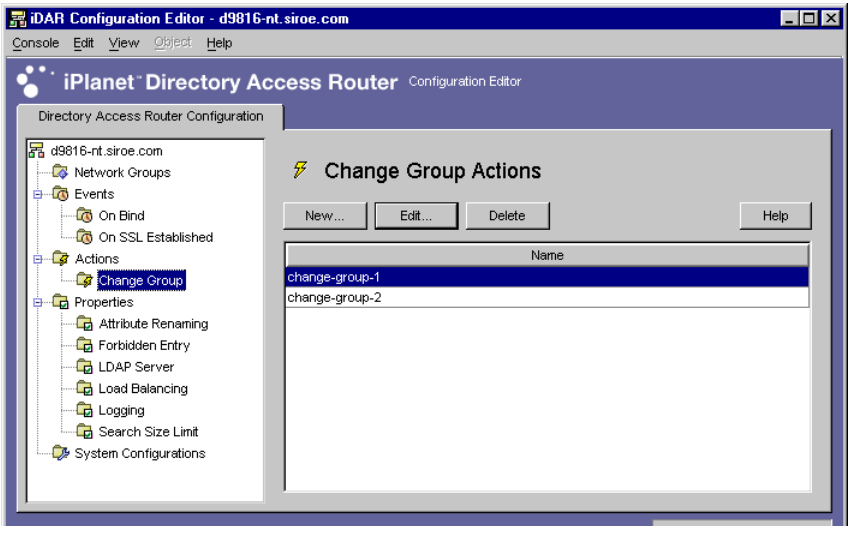
Creating Action Objects

You can create objects for actions that needs to be executed when certain events occur. The instructions below explain how to create an action object for changing groups.

To create an action object for a client to change from one group to another:

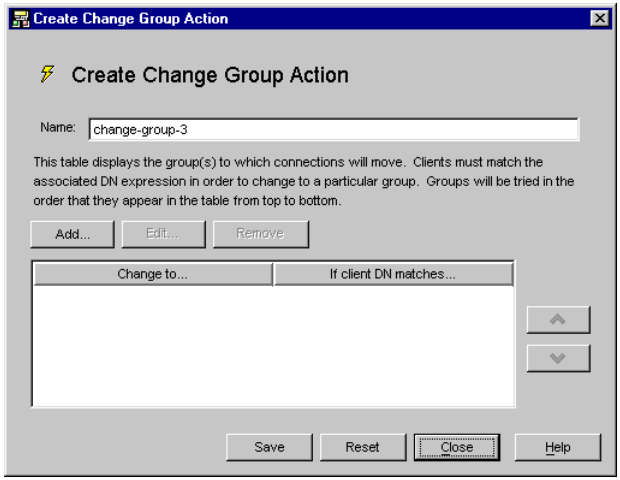
1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, expand the Actions node, and then, select Change Group.

   The right pane shows the list of existing action objects.



3. Click New.

   The Create Change Group Action window appears.



4. In the Name field, type a name for the object. The name must be a unique alphanumeric string.

5. In the Action tab, select the action to be performed when the event occurs (that is, when the event evaluates to TRUE).

   Change Table. Displays a list of groups to which a client can change. For a change to occur, the client must match a DN expression associated with each group. To edit the DN expression associated with a particular group or "no change" entry, click the "If client DN matches" column in the table. The list is evaluated from top to bottom until a DN expression is matched. Therefore, it is important that the most general DN expression is at the bottom of the list so that all expressions will be evaluated.

   Regular expressions must be normalized, that is, there should be no spaces in between RDN components and the equal to (=) sign, all attribute names and values must be capitalized.

   You can use the following book as a reference on regular expressions: *Mastering Regular Expressions*, by Friedl and Oram, published by O'Reilly, ISBN: 1565922573.

   Add. Displays a menu for adding a group to which a client connection could potentially change. Group change entries can be of the following types: "Group change entry," or "No change entry."

Group change entry. Displays a dialog for selecting a network group to which a client will change depending on whether or not the associated DN expression evaluates to TRUE.

No change entry. Adds a row to the table indicating that NO change should occur if the associated DN expression evaluates to TRUE. This is useful in providing a "short circuit" of the evaluation of the change group list.

Edit. Displays a dialog for editing the currently-selected entry in the table.

Remove. Removes the currently-selected entry in the table.

6. Click Save to create the action object.

   The iDAR configuration is modified, and you're prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

7. Repeat Step 3 through Step 6 to create any additional objects.

8. Restart the servers; see "Restarting iDAR," on page 57.

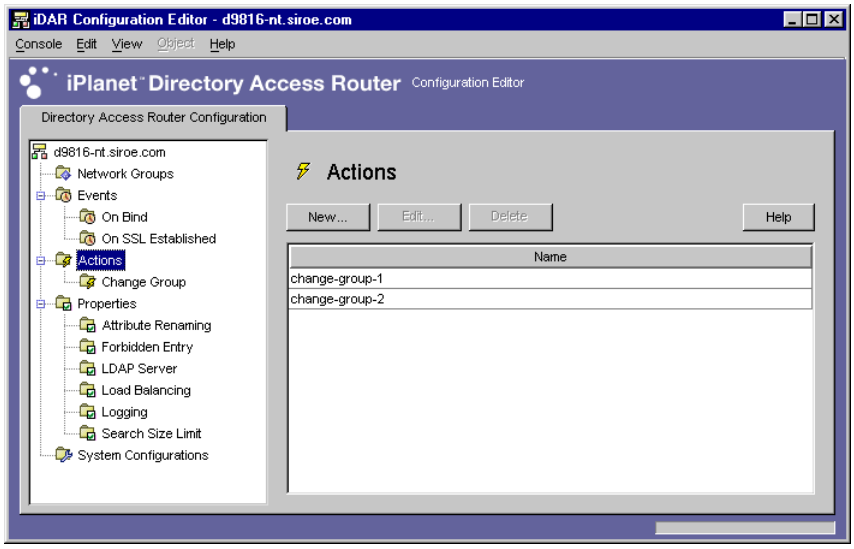
Modifying Action Objects

To modify an action object:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

2. In the navigation tree, select Actions.

   The right pane shows the list of existing action objects.



3. In the list, select the action object you want to modify and click Edit.
4. Make the required modifications.
5. Click Save to save your changes.

   The iDAR configuration is modified, and you are prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.
6. Repeat Step 3 through Step 5 to modify any additional objects.
7. Restart the servers; see "Restarting iDAR," on page 57.

Deleting Action Objects

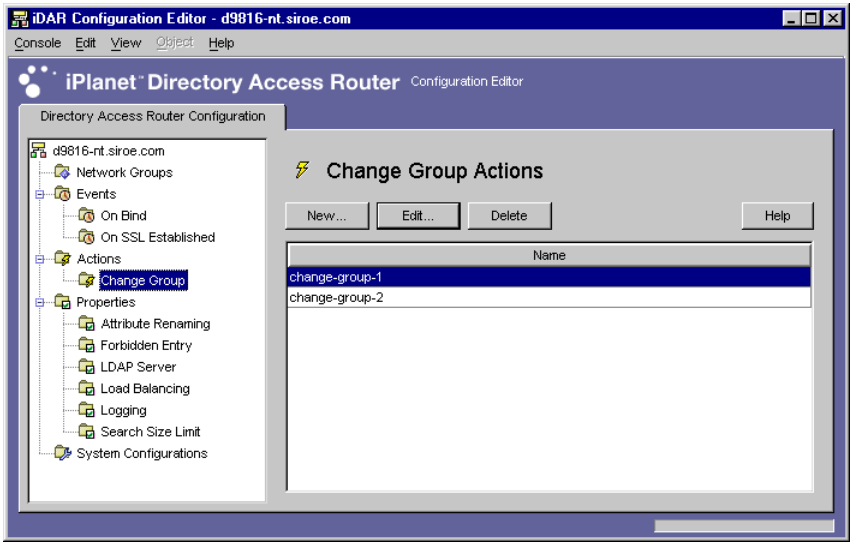
You can delete any unwanted action objects from the iDAR configuration. Before deleting an action object, make sure that it's not used in the configuration of any event objects.

To delete an action object:

1. Access the iDAR Configuration Editor Console; see "Accessing the iDAR Consoles," on page 46.

2. In the navigation tree, select Actions.

   The right pane shows the list of existing action objects.



3. In the list, select the action you want to delete and click Delete.

4. Confirm your action.

   The name of the object you deleted is now removed from the list. The iDAR configuration is modified, and you are prompted to restart the servers that rely on this configuration. Don't restart the servers yet. You can do this after you've completed all the configuration changes.

5. Repeat Step 3 and Step 4 to delete any additional objects.

6. Restart the servers; see "Restarting iDAR," on page 57.

Chapter 10

Configuring and Monitoring Logs

This chapter explains how to configure iPlanet Directory Access Router (iDAR) to log entries or messages and then monitor its activities with the help of the logged entries using the iDAR Server Console.

The chapter has the following sections:

Overview of Logging

iDAR can maintain two types of logs:

- System Log
- Audit Log

The sections that follow explain both in detail.

System Log

iDAR can maintain extensive log records of various events and system errors so that you can monitor and debug the system. All log records can be maintained in text files and can be stored in your local file system for quick and easy retrieval. By default, iDAR writes log entries to this file:

```
<server-root>/idar-<hostname>/logs/fwd.log
```

Each message in the log file is time stamped. It also has the process number and a message number that is internal to iDAR.

For identification and filtering purposes, events logged by iDAR are classified into various categories. These are listed in Table 10-1. Each category represents messages that are of the same or a similar nature or that belong to a specific functional area. Based on the configuration, a log file can record entries that fall under one or more of these categories.

In the iDAR configuration, each message category corresponds to a specific log level. Log levels indicate the level of logging to be performed by the server—that is, how detailed the logging should be.

- A higher priority level means less detail because only events of high priority are logged.
- A lower priority level means greater detail because more kinds of events are recorded in the log file.

Table 10-1 lists the message categories in the descending order of priority—`Critical` has the highest priority level and `Detailed trace` has the lowest priority level.

Table 10-1 Log Levels

Log Level or Severity	Description
Mandatory	Mandatory messages are those that are always printed to the log. These messages indicate the configuration that iDAR read, iDAR version number on startup, etc. Messages pertaining to this level is not configurable.
Critical	These messages indicate iDAR encountered some problems that needs immediate attention. For example, *iDAR process 1234 has exited, attempting restart in 10 seconds*.
Exception	These messages indicate unexpected error conditions, such as an incorrectly formatted LDAP message received from client / server by iDAR. For example, *Could not decode search request*.
Warning	These messages specify error conditions that iDAR can ignore but must be investigated by the administrator. For example, *Local host name lookup failed. System default group may not function correctly*.
Notice	These messages are informational. For example, *Received NULL continuation reference from server. Discarding...*

Table 10-1 Log Levels *(Continued)*

Log Level or Severity	Description
Trace	These are debug messages. For example, *Result received from server lderr =32, matched=o=sun.com, errtxt=no such object.*
Detailed trace	These messages provide more detailed debugging information such as Requested Anonymous bind for recycling connection. These messages usually have meaning for the iDAR engineering / support team.

iDAR enables you to specify the amount of logging—you can use log levels to filter log entries based on the severity of an event. By default, the level is set to `Warning`.

NOTE The log level is additive; that is, if you choose `Warning` as the log level, `Warning`, `Exception`, and `Critical` level messages will be logged. Log data can be voluminous, especially at lower (more verbose) logging levels. Make sure that the host machine has sufficient disk space for all the log files.

Optionally, on platforms other than Windows NT, you can configure iDAR to send log messages to the `syslog` daemon instead of a file; you cannot send the log messages to both a file and to the `syslog` daemon at the same time. If you opt for this configuration, make sure that your `syslogd` is properly configured. For example, to have all the messages written to a particular file `/var/adm/messages` the following line must be added to the file `/etc/syslog.conf`:

```
daemon.crit;daemon.warning;daemon.info;daemon.debug
/var/adm/messages
```

Note that iDAR uses the `daemon` facility, with the `crit`, `warning`, `info`, and `debug` priorities or log levels. Table 10-1 shows the mapping between syslog events and iDAR events.

Table 10-2 Mapping of Log Levels

iDAR Event	syslog Event
Mandatory	info
Critical	crit
Exception	err
Warning	warning

Table 10-2 Mapping of Log Levels *(Continued)*

iDAR Event	syslog Event
Notice	info
Trace	info
Detailed trace	info

iDAR does not rotate logs. You need to do this manually and it will usually require a restart. If you can afford to lose a few messages, on the UNIX platforms, you can rotate logs with the following sequence of commands.

```
cp fwd.log /logvol/fwd.log.old
cp /dev/null fwd.log
```

By default, iDAR does not delete any of the log files automatically. Because the log files are saved in your local file system, these files eventually take up a considerable amount of disk space. You can avoid this problem by manually deleting the log files from the local file system. If you want to keep specific log files for future use, be sure to archive or back them up.

Audit Log

In addition to logging system and error messages, iDAR can also maintain audit trails for all events and connection statistics—for example, the DN of a client that just completed a bind / unbind with an LDAP directory can be logged.

By default, iDAR is not configured to log audit messages. You can enable this feature at any time. You can also specify whether to log audit messages to the same file to which system log entries are written or to an alternate file. Unless configured to write to a different file, the audit messages (along with the other log messages) are logged to the same file to which system log entries are written; for details, see “System Log,” on page 153.

NOTE Audit records enable you to detect any unauthorized access or activity. It’s recommended that you enable this feature. Also, as a security measure, you should periodically examine the iDAR audit log for any unusual activity.

Configuring Logs

To configure iDAR to log entries, follow these steps:

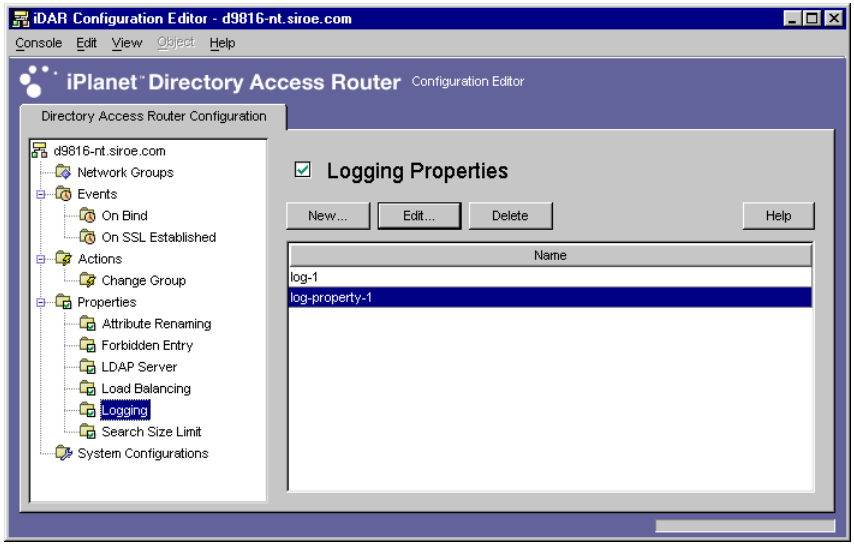
- Step 1. Define a Logging Property
- Step 2. Specify the Logging Property to Use

Step 1. Define a Logging Property

This step is required only if you want to create or define an object for the Log Property. If you have already created objects for the log property and want to use one of them, then skip to the next step.

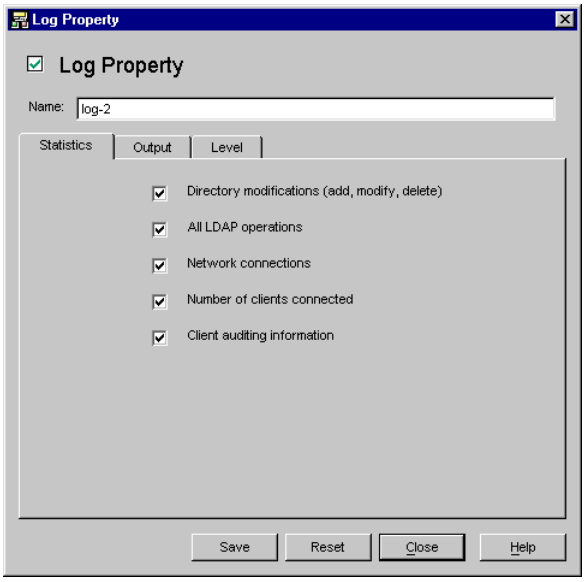
1. Access the iDAR Configuration Editor Console; see “Accessing the iDAR Consoles,” on page 46.
2. In the navigation tree, expand the Properties node, and then select Logging.

   The right pane shows the list of existing objects for the logging property.



3. Click New to define a new object.

   The Log Property window appears.



4. In the Name field, type a name for the object. The name must be a unique alphanumeric string.

5. In the Statistics tab, specify the kind of information to be logged.

   Check the boxes referring to the type of logging messages desired. By default none of the options are selected. Log messages are classified into the following groups: directory modifications, all LDAP operations, network connections, number of clients connected, and client auditing information.

   Directory modifications. Statistics about operations that write to the directory, like add, modify, and delete will be logged.

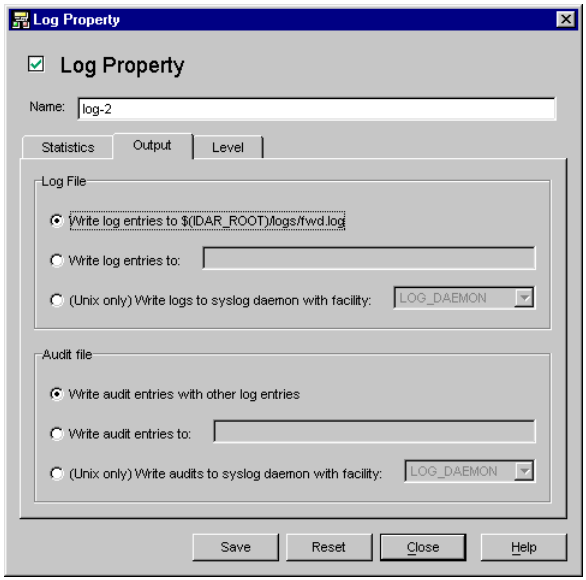
   All LDAP operations. Statistics about all LDAP operations will be logged.

   Network connections. Statistics about network connections will be logged.

   Number of clients connected. General statistics such as how many clients are connected will be logged.

   Client auditing information. Audit information such as the DN of client that just completed a bind / unbind will be logged.

6. Select the Output tab and specify where log entries should be sent and whether to log audit traces.



Log file. Displays options governing where iDAR will write its log entries.

Write log entries to $(IDAR_ROOT)/logs/fwd.log. This is the default setting in which iDAR will write its log entries to the file `$(IDAR_ROOT)/logs/fwd.log` where `$(IDAR_ROOT)` is the directory under the server root where iDAR is installed, typically `/usr/iplanet/servers/idar-<hostname>` or `\Program\Files\iPlanet\Servers\idar-<hostname>`.

Write log entries to. Specify an alternative file to which iDAR will direct its log entries. The file separator must follow UNIX conventions regardless of platform.

Write logs to syslog daemon with facility. (UNIX only) Choose a syslog facility code that iDAR will use to log entries. This setting should only be chosen if this log property is to be used by iDAR server installed on a UNIX machine. Specifying this option for iDAR installed on a Windows NT system will make it inoperable. It is recommended that if you would like to specify values for this attribute, you should create separate log properties for Windows NT and UNIX.

Audit File. Displays options governing where iDAR will write its audit log entries. For this feature to work, audit logging must be enabled by selecting the “Client auditing information” option in the Statistics tab.

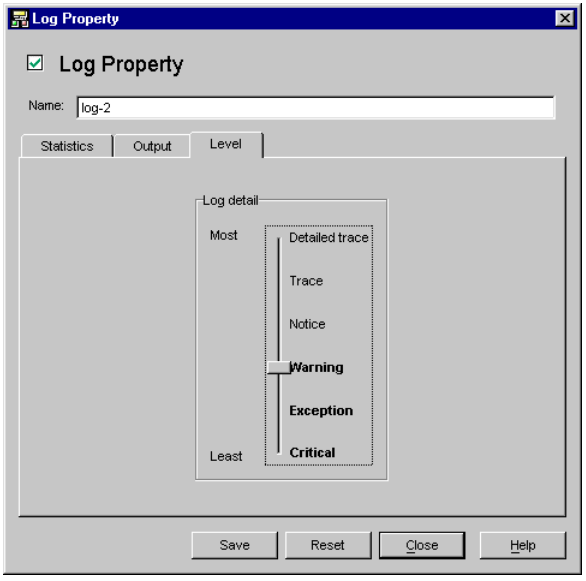
Write audit entries with other log entries. This is the default setting in which iDAR will write its audit log entries to the same output specified in the log file settings above.

Write log entries to. Specify an alternative file to which iDAR will direct its audit log entries. The file separator should follow UNIX conventions regardless of platform.

Write audits to syslog daemon with facility. (UNIX only) Choose a syslog facility code that iDAR will use to log audit entries. This setting should only be chosen if this log property is to be used by iDAR servers hosted on a UNIX machine. Specifying this option causes a Windows NT based iDAR to become inoperable. It is recommended that if you would like to specify values for this attribute, you should create separate Log Property objects for Windows NT and UNIX.

7. Select the Level tab and specify the log level—the amount of logging detail desired.

   Move the slider up to indicate more detail, or down to indicate less detail.



8. Click Save to save your changes.

   The name of the object now appears in the list. The iDAR configuration is modified, and you're prompted to restart the server.

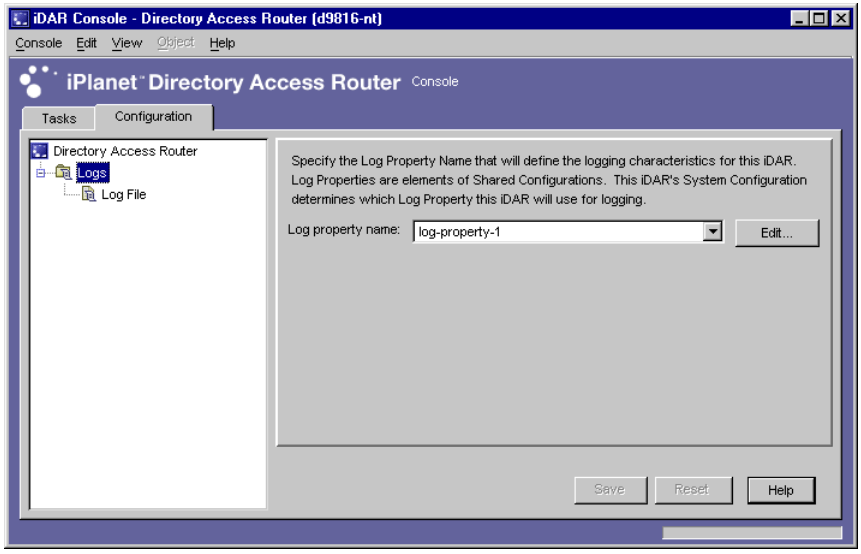
9. Restart the server; see "Restarting iDAR," on page 57.

Step 2. Specify the Logging Property to Use

In this step, you select an existing log property to be used for logging messages.

1. Access the iDAR Server Console; see “Accessing the iDAR Consoles,” on page 46.
2. Select the Configuration tab, and then, in the navigation tree, select Logs.

   The right pane shows information regarding the log property specified by the current system property.



3. In the “Log property name” drop-down list, select the property you want to use.

   To verify that you’re using the correct property, click Edit and take a look at the configuration. If required, you can make any configuration changes.

4. Click Save to save your changes.

   iDAR is now configured to log messages as defined in the configuration. The iDAR configuration is modified, and you are prompted to restart the server.

5. Select the Tasks tab and restart the server; see “Restarting iDAR,” on page 57.

Monitoring Logs From iDAR Server Console

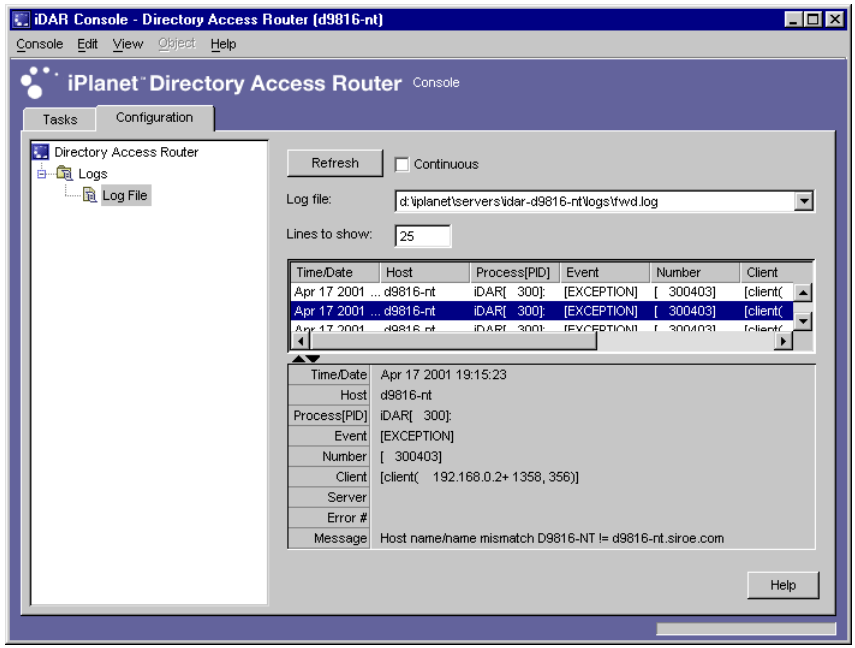
Once you've configured iDAR to log messages (see "Configuring Logs," on page 157), you can monitor its activities by viewing the log messages. For example, when you have problems with iDAR that require troubleshooting, you may find it helpful to check the error or informational messages that the server has logged. Also, by examining the log files you can monitor many aspects of iDAR's operation.

To facilitate this, the iDAR Server Console provides a simple mechanism for viewing the contents of log files. The contents of the log file you choose to view are displayed in the form of a table. The table is split; the top pane shows log records in tabular format and the bottom pane shows the currently-selected record in detail. Each log record contains information such as the date and time the message was logged, the severity of the message, and a general description of the log.

Once you open a log file for viewing, you can read its contents partially by specifying the number of records or entries to be displayed. The instructions below explain how to view log records in a file:

1. Access the iDAR Server Console; see "Accessing the iDAR Consoles," on page 46.
2. Select the Configuration tab, and then, in the navigation tree, expand Logs.
3. Select Log File.

   The right pane shows viewing options for entries logged to a file. You can select any of the log files specified in the current log property; iDAR can contain separate files for logging and auditing information, if configured to do so.



Description of the form elements are as follows:

Refresh. Reads the log and displays the records in the table below.

Continuous. Select this setting to have this view continuously refreshed with the most current log records.

Log file. Displays the name of the file currently being viewed.

Lines to show. Specifies the maximum number of lines to read from the log file.

Part 3

Command-Line Configuration

Chapter 11

Configuration Overview

This chapter provides an overview of iPlanet Directory Access Router (iDAR) configuration via the command line. The chapter contains the following sections:

Introduction

Configuration of iDAR is a two step process. First, a startup configuration file (`tailor.txt`) must be created with information that will enable iDAR to find its configuration; for details, see “Startup Configuration File,” on page 192. Then, the iDAR configuration file (`tailor.ldif`) must be created; it must reside in another file on disk or in an LDAP directory server. In the latter case, the startup configuration must also contain information about what mechanism is used to bind to the directory server. The remainder of this chapter is an overview of the iDAR configuration.

iDAR Configuration File Format

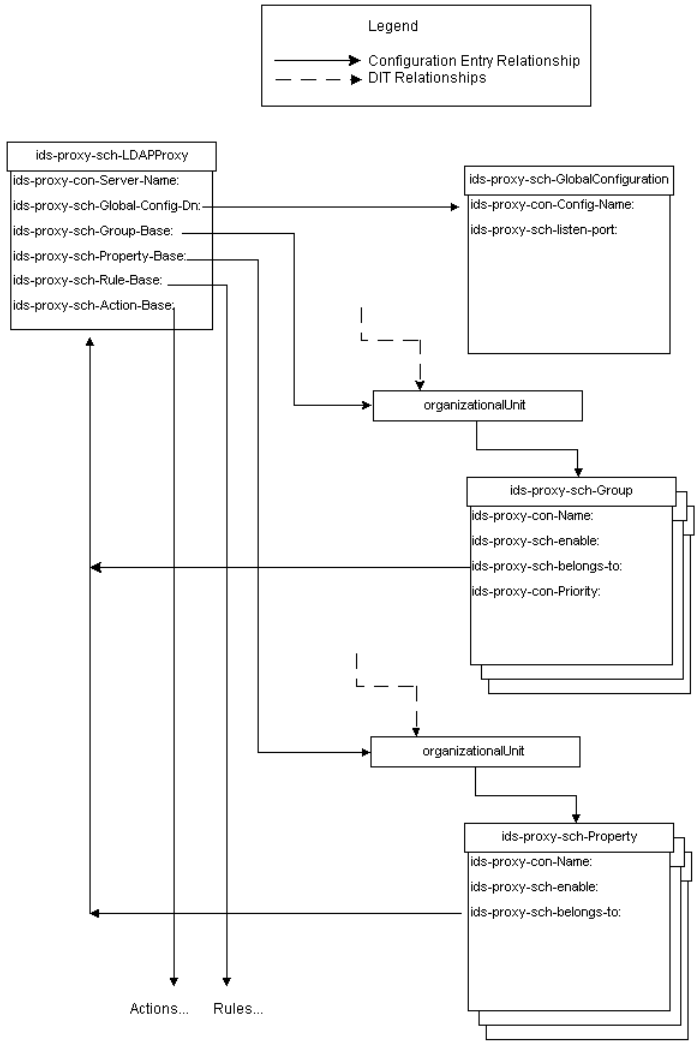
The iDAR configuration file (`tailor.ldif`) is in LDAP Data Interchange Format (LDIF). LDIF is used to represent LDAP entries in a simple text format. This format is used for iDAR configuration files for two reasons.

First, a simple text file is easy to modify, and second, using an LDIF file allows you to store the iDAR configuration in an LDAP directory server. This may be important for customers who decide to distribute their LDAP infrastructure geographically but want to manage the administration of that infrastructure centrally. By choosing to keep all iDAR configurations in a central LDAP repository, central administration of the iDARs can be achieved. The configuration file is divided into a number of distinct sections, each potentially comprising multiple directory entries:

- *Server* - The server entry identifies the iDAR server instance. It is constructed by including the `ids-proxy-sch-LDAPProxy` object class. The startup configuration file points to this entry. The server entry has some basic attributes that identify where the global configuration object is located, the name of the server instance, and the DNs under which the group, property, rule, and action object entries reside. Figure 11-1 on page 169 shows these relationships graphically.
- *Global* - The global configuration entry is constructed of attributes that identify those iDAR properties that are common to all clients. Global entry is constructed by including the `ids-proxy-sch-GlobalConfiguration` object class. See Chapter 15, “Global Configuration.”
- *Groups* - Each group entry identifies a specific client community, and specifies the restrictions to enforce on clients that match that group. Group entries are constructed by including the `ids-proxy-sch-Group` and the `ids-proxy-sch-NetworkGroup` object classes. See Chapter 16, “Groups Configuration.”
- *Properties* - Each group entry may include a set of properties defined by property entries. These property entries describe more specialized restrictions on the clients. Property entries are constructed by including the `ids-proxy-sch-Property` object class together with its associated object classes. Properties are included within group entries by defining the property entry name as the value of the `ids-proxy-con-include-property` attribute. See Chapter 17, “Properties Configuration.”
- *Rules* - A group or property entry may include one or more rule entries. These rules are evaluated at predetermined points of execution (determined by the rule type), and one or more actions (defined separately) can be taken based on the result of the evaluation. Rule entries are constructed by including the `ids-proxy-sch-Rule` object class together with its associated object classes. Rules are included within group entries by defining the rule entry name as the value of the `ids-proxy-con-include-rule` attribute. See Chapter 18, “Events Configuration.”

- *Actions* - Depending on the evaluation of a rule, one or more actions may get executed. Action entries specify what the iDAR needs to do in the case where the rule that includes them evaluates to `TRUE`. Action entries are constructed by including the `ids-proxy-sch-Action` object class together with associated object classes. Actions are included as part of rule entries by defining the action entry name as part of the value of the `ids-proxy-con-execute` attribute. See Chapter 19, “Actions Configuration.”

Figure 11-1 Information Model and Configuration Entry Relationship



It is worth noting at this point that the iDAR configuration file adheres to the following naming convention:

Those elements of the configuration file that specifically relate to the schema structure (as opposed to the iDAR configuration elements) start with the prefix `ids-proxy-sch` where the `sch` component indicates the schema. Examples of such elements are names of object classes (e.g., `ids-proxy-sch-Group`) and attributes with a syntax of DN (e.g., `ids-proxy-sch-Property-Base`). Those elements of the configuration file that are specifically related to the iDAR configuration start with the prefix `ids-proxy-con` where the `con` component indicates configuration. Almost all attributes defined as part of the iDAR schema (other than those with DN syntax) start with `ids-proxy-con` (e.g., `ids-proxy-con-listen-port`). This is because the vast majority of attributes relate to iDAR configuration options. The two notable exceptions are `ids-proxy-sch-Enable` and `ids-proxy-sch-belongs-to`. The first attribute determines if the entry is part of the configuration or not and the second attribute determines which server configuration(s) the entry belongs to.

Building a Configuration File

To build a configuration file:

- Define Global Entries
- Define Property Entries
- Define Action Entries
- Define Rule Entries
- Define Group Entries

Define Global Entries

The configuration file is constructed by first defining a set of global entries that represent those properties of iDAR that are common to all client groups.

Examples of such properties are:

- The host name and port number iDAR is going to listen on for incoming connections
- The maximum number of simultaneous clients connections accepted

- SSL / TLS configuration
- The search base for the group, property, rule and action configuration entries

A complete description of all the global entry object classes and attributes can be found in Chapter 15, “Global Configuration.”

Define Property Entries

Next, the property entries should be defined for all properties to be utilized within the configuration. For example, if iDAR is going to be configured for load balancing and automatic failover and failback, then a load balancing entry should be defined using the `ids-proxy-sch-LoadBalanceProperty` object class. A complete description of all the property entry object classes and attributes can be found in Chapter 17, “Properties Configuration.”

Define Action Entries

The action entries should be defined next. iDAR currently supports the `ids-proxy-sch-ChangeGroupAction` object class that allows iDAR to be configured to change a client from one access group to another based on the evaluation of a rule. This is typically used to support mobile users where the rule might be `ids-proxy-sch-OnBindSuccessRule`. A mobile user could connect to the iDAR with a dynamic IP address and drop into a “default” access group. The “default” access group would have the `ids-proxy-sch-OnBindSuccessRule` rule specified that only evaluates to `TRUE` if the bind credentials provided by the mobile user are authenticated. This rule would also have the `ids-proxy-sch-ChangeGroupAction` specified to change the mobile user's access group from the “default” to the access group the mobile user is usually assigned to when accessing iDAR with a static IP address. Actions need to be defined before rules since the rule definition requires you to specify a previously defined action. A complete description of all the action entry object classes and attributes can be found in Chapter 19, “Actions Configuration.”

Define Rule Entries

Rule entries should be defined next, following the definition of the action entries. iDAR currently supports the `ids-proxy-sch-OnBindSuccessRule` and the `ids-proxy-sch-OnSSLEstablishedRule` object classes. If the iDAR administrator wishes to change the access group of a client based on the evaluation of one of

these two rules then both an `ids-proxy-sch-ChangeGroupAction` action entry and one of the two rule object classes need to be defined. The rule entry must include the name of the action entry as part of the value of the `ids-proxy-con-execute` attribute. A complete description of all the rule entry object classes and attributes can be found in Chapter 18, “Events Configuration.”

Define Group Entries

Having defined global, property, action and rule entries you are now at the point to define the group entries. Group entries are the core entries of the iDAR configuration file because they identify the different LDAP client groups and determine the routing, access controls, request filtering, response filtering, etc., that iDAR applies to LDAP operations initiated by those clients. There are currently over 40 different attributes for configuration options within the `ids-proxy-sch-Group` and `ids-proxy-sch-NetworkGroup` object classes. These can be classified into the following categories:

- Group object attributes
- Client identification
- Including rules and properties
- SSL policy
- LDAP bind request validation
- Control of forwarding different LDAP operation types
- Controlling search, compare and modify requests and responses
- Referral following policy
- Controlling the directory server load
- Hiding a subtree of entries

A complete description of all the group entry object classes and attributes can be found in Chapter 16, “Groups Configuration.”

Group entries are constructed by first considering the client communities you need to support or differentiate between. Identify the clients making up a community and create a group for them by first identifying them in the group entry by correctly defining the `ids-proxy-con-client` multi-valued attribute. If you have previously defined property and rule entries that are appropriate for this group,

then include them by configuring the `ids-proxy-con-include-property` and/or the `ids-proxy-con-include-rule` attributes. Using the attributes defined for the group object classes, configure additional controls for this group as appropriate. Then move on to the next group definition.

Configuration File Build Tool

To help iDAR administrators get started with configuring iDAR, iDAR ships with a command-line configuration tool that allows you to build a number of different configuration files. A complete description of the iDAR configuration file build tool can be found in Chapter 13, “Configuration Generation Tool.”

The command-line tool, `tailor` allows you to specify command line parameters that the tool uses to automatically generate a working iDAR configuration file in LDIF format. Note that there are limitations to what the command-line tool can generate so in most cases it should be considered a tool for building a basic working configuration file that can then be modified to provide further customization.

Currently, the build tool supports generating configuration files for the following customer scenarios:

- The bare minimum, “out-of-the-box” configuration to get the user up and running with a working iDAR configuration immediately. This configuration is a simple “passthru” from the LDAP client to a single LDAP directory server via iDAR. There is no filtering of requests or responses and no additional access controls. Only one client group will be generated.
- A more sophisticated configuration that incorporates load balancing and automatic failover and failback into the basic configuration mentioned above. The user can specify any number of LDAP directory servers and load balance the client load across them all. Only one client group will be generated.
- The third scenario is probably the most complex. This scenario will allow the LDAP client to bind to the LDAP directory server either anonymously or with bind credentials and treat the two differently. The `tailor.tcl` tool will generate two groups: one that is employed when a client binds with bind credentials and another group that acts as a “default” group. All clients that bind anonymously will fall into the “default” group and will be limited to search operations only. Those clients that bind with bind credentials will initially drop into the “default” group. The “default” group has the `ids-proxy-sch-OnBindSuccessRule` rule included in it. This rule has the `ids-proxy-sch-ChangeGroupAction` action associated with it. So clients that bind with bind credentials that are subsequently authenticated cause the rule

to evaluate to TRUE and the action to get executed that changes the group the client belongs to from the "default" group to a group with more access rights. The concept behind these two groups is to grant more access rights to those clients binding with bind credentials and to restrict the access rights of clients binding anonymously.

The tailor tool can be helpful in generating LDIF configuration files. For example, tailor could be used to generate a configuration file for the typical customer deployment of an internal high availability configuration described using Figure 2-1 on page 30. You would use tailor load balancing scenario with the following syntax:

```
tailor     -server ldaphost-1 -server ldaphost-2 -server ldaphost-3\
           -server ldaphost-4
```

No further customization of the configuration file generated by tailor is necessary. The firewall deployed in this scenario should ensure there is no external access to either the iDAR or the LDAP directory server. However, for additional peace of mind the user could customize the configuration file by identifying "internal" LDAP clients via the ids-proxy-con-Client multi-valued attribute in the ids-proxy-sch-NetworkGroup object class. Any client that attempts to connect to the iDAR but hasn't been identified by the ids-proxy-con-Client multi-valued attribute will be rejected.

Don't forget to use TCP wrapper or equivalent program on UNIX platforms and TCP/IP port filtering on Windows NT to ensure access to the LDAP directory servers is only permitted for the iDARs you deploy.

iDAR Decision Functions

This section describes the flow of control in iDAR for some specific functionalities.

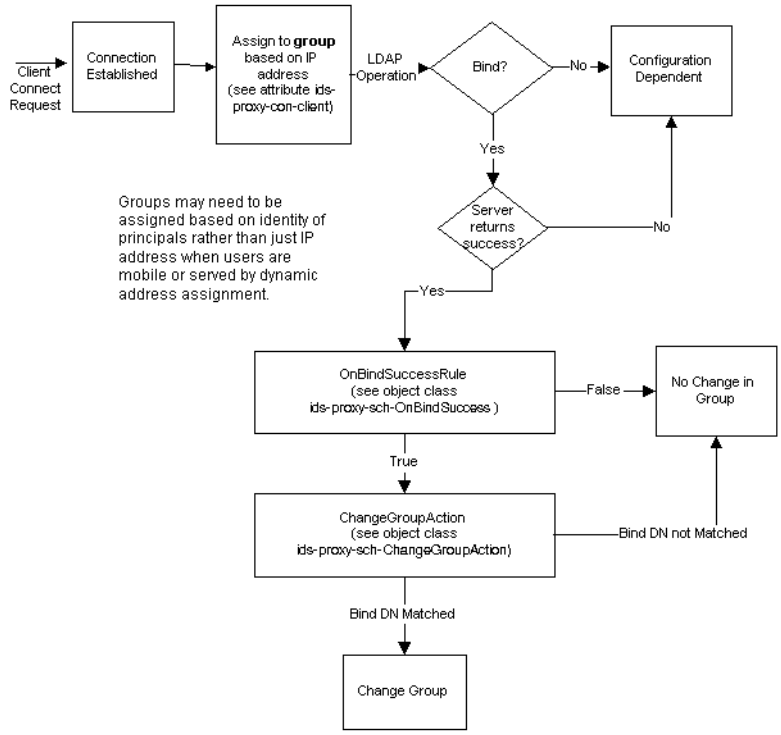
Establishing Group on Connection

When a client makes a connection to iDAR, it checks the ids-proxy-con-Client attribute in ids-proxy-sch-NetworkGroup object entries until it finds a match. The ids-proxy-sch-NetworkGroup objects are tried in highest to lowest priority defined by the ids-proxy-con-priority attribute. iDAR places the client in the first group whose ids-proxy-con-client attribute matches the IP address of the client. If no matching groups are found, the connection is closed.

Change Group on Bind

When the client initially connects, it is placed in a group based on its IP address. The client can be moved to a different group with different access controls when it binds to a directory. To accomplish this, the initial group object must include a rule object that is evaluated on a successful bind operation. If the rule evaluates to TRUE, the change group action is taken to move the client to a different group. Figure 11-2 illustrates this functionality.

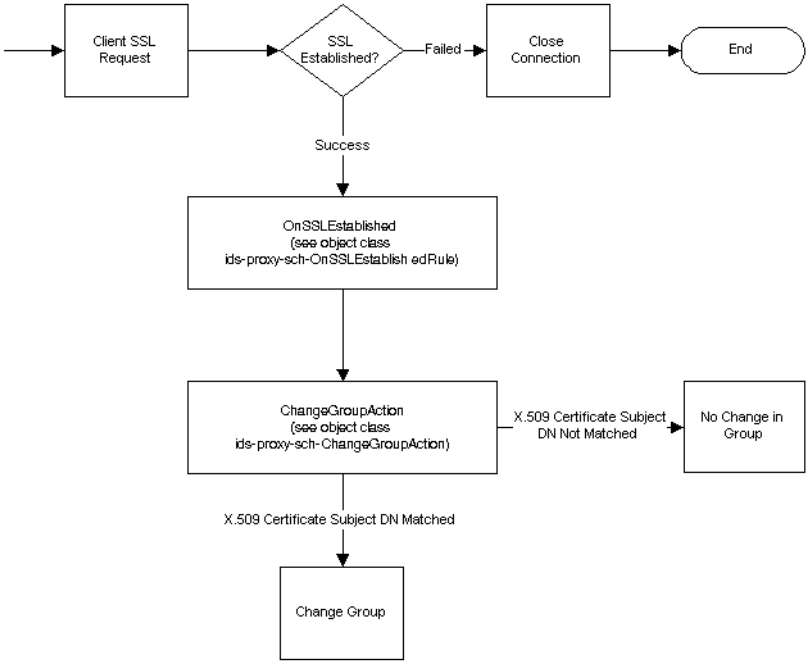
Figure 11-2 Change Group on Bind



Change Group on Establishment of TLS

Similar to the change group on bind mechanism is the change group on establishment of TLS, whereby a client can change group when it successfully establishes an TLS session. The SSL Established rule is evaluated when the client establishes TLS, following which the Change Group action follows. This functionality is illustrated in Figure 11-3.

Figure 11-3 Change Group on Establishment of TLS



High Availability Setup

If you have configured more than one backend directory server, then you can set up iDAR to load balance across these and fail over to another if one of the backend servers go down. In order to do this, you must create a Load Balance Property (see “Load Balancing Property,” on page 126 or “ids-proxy-sch-LoadBalanceProperty Object Class,” on page 226) and include it in the group object for which you want to load balance. You will also need to create LDAP Server Properties (see “LDAP Server Property,” on page 121 or “ids-proxy-sch-LDAPServer Object Class,” on page 231) for each of your backend servers and include it in the Load Balance Property. You must specify the amount of load in percentage of total load each of your backend servers should handle in the Load Balance Property object. With this setup, iDAR will redistribute load across its backend directory servers if one of

them goes down. It will fail over clients from one server to another in case the first goes down. iDAR will also fail over if the network link between itself and the LDAP server is down or if the LDAP server becomes unresponsive (see “ids-proxy-con-keepalive-interval,” on page 233).

NOTE iDAR is unable to fail over if the client was bound using a SASL mechanism.

Following Referrals

iDAR can be set up to follow referrals for LDAPv2 clients that cannot do so on their own. Your backend LDAP directory server must be capable of sending referrals, i.e., it must support LDAP v3 standards. Configure iDAR to use LDAP v3 between itself and the backend LDAP server in order for iDAR to receive referrals from the directory server (see “ids-proxy-con-use-version,” on page 231). Then set your group’s referral (see “Controlling the Return of Referrals,” on page 221) and continuation referral policy (see “ids-proxy-con-search-reference,” on page 221).

Chapter 12

Configuring Security

iPlanet Directory Access Router (iDAR) supports SSL/TLS for secure communication between its clients and backend directory servers. It also supports SASL bind mechanisms for client authentication. The following sections describe how to configure SSL/TLS functionality in iDAR from the command-line and what types of SASL mechanisms are supported:

Configuring TLS/SSL in iDAR

Transport Layer Security (TLS) is the Internet standard protocol for secure communication between two end points. TLS/SSL is widely used with the HTTP protocol on the Web. Because of US government restrictions on export of cryptography, TLS is supported in the following ways in iDAR:

- For US, Canadian, and customers in specific industry segments, iDAR supports 168-bit encryption.
- iDAR supports 56-bit encryption for export purposes to all other customers.

iDAR supports TLS/SSL for encryption purposes only. An encrypted TLS/SSL connection between clients and iDAR will protect the data transmitted over that connection. iDAR can also be configured to always create a SSL session between itself and the backend directory server if the network between itself and the backend LDAP server is not safe, and the directory server supports TLS/SSL.

iDAR cannot act as a “passthru” for an “end-to-end” TLS/SSL connection between the client and a backend directory server because iDAR has to be configured to bind to a directory server with one set of bind credentials and does not use the bind credentials presented by the client. Furthermore, a compliant TLS/SSL implementation would treat the iDAR as performing a “man-in-the-middle” attack, since proxying a client’s credentials is not allowed in strong authentication.

iDAR can be configured to establish TLS/SSL between itself and the client and itself and the server in many different combinations. It supports both the Start TLS method where clients establish TLS/SSL session over the standard LDAP port and the alternate port 636 method where clients connect to an alternate secure port if they want to establish TLS/SSL.

The following matrix summarizes the possible configuration scenarios. The columns represent connections between client and iDAR, and the rows represent connections between iDAR and the backend LDAP directory server.

Table 12-1 TLS/SSL Configuration Scenarios

	iDAR	CLIENT		
		No TLS	TLS using alternate port 636	TLS using START TLS mechanism
SERVER	No TLS	X	X	X
	TLS using alternate port 636	X	X	X
	TLS using START TLS mechanism	X	X	X

Steps to Configure TLS/SSL Support

1. Generate a certificate request and a private key for iDAR using the supported `certreq` utility (see “Generating a TLS Key Pair,” on page 182).
2. Send the proxy server’s certificate request generated by `certreq` to a registered Certification Authority (CA) of your choice, e.g., Thawte, and obtain a certificate from the CA.
3. In the `ids-proxy-sch-GlobalConfiguration` object entry, set the `ids-proxy-con-ldaps-port` attribute to the port number on which to listen for LDAPS (LDAP over TLS/SSL) connections. This is optional and is necessary for clients that use the alternative port 636 method to establish TLS/SSL.

4. Make the following changes to the `ids-proxy-con-GlobalConfiguration` object entry.

   a. Set the `ids-proxy-con-ssl-cert` attribute to the location path name of the file on disk containing the server's own certificate signed by the CA.

   b. If you are configuring iDAR to communicate with a directory server over TLS/SSL and the directory server in question supports receiving certificates, set the `ids-proxy-con-send-cert-as-client` attribute to `TRUE`. Otherwise allow it to default to `FALSE`.

   c. Set the version of TLS that iDAR will use for establishing TLS with its clients or the backend directory server by setting the value of `ids-proxy-con-server-ssl-version` and/or `ids-proxy-con-client-ssl-version` to the appropriate version number. (see “ids-proxy-con-server-ssl-version, ids-proxy-con-client-ssl-version,” on page 206). The defaults interoperate with iPlanet Directory products.

   d. If you want iDAR to verify the certificates presented to it you must set the `ids-proxy-con-ssl-cafile` to the file containing the root certificates of CA's that iDAR can trust. This file must contain the list of certificates in PEM format.

   e. If you want your clients to always present a certificate when they initiate a TLS session with iDAR, set `ids-proxy-con-ssl-cert-required` attribute to `TRUE`.

5. For each group definition configured, set the `ids-proxy-con-ssl-policy` attribute in the `ids-proxy-sch-NetworkGroup` object entry as appropriate depending on whether you want to force the client to start a TLS session before sending any LDAP operation, leave the decision to the client, or disallow the client to start a TLS session. See “ids-proxy-con-ssl-policy,” on page 212.

6. Make the following changes to each of the `ids-proxy-sch-LDAPServer` object entries.

   a. Set `ids-proxy-con-link-security-policy` to an appropriate value (see “ids-proxy-con-link-security-policy,” on page 232) so that iDAR will always establish SSL/TLS to the backend server, never establish TLS/SSL to the backend server, or only establish SSL/TLS with the backend server when the client does the same to iDAR.

   b. Set `ids-proxy-con-x509cert-subject` to the DN of the backend server that iDAR must receive as value of the `subject` attribute in the X509 certificate it will receive from the backend server. iDAR will accept any name if this attribute is not set.

Generating a TLS Key Pair

In order for TLS to operate, the server must have a private key and a certificate. The `certreq` program provided in the `<server-root>/bin/idar/server/bin` directory can be used to generate a private key and a certificate request for the server.

In order to have iDAR's certificate certified by a Certification Authority (CA), you should send the server's certificate request generated by `certreq` to the CA, and receive back a certificate for the server signed by the CA.

Generating Files With certreq

The `certreq` program is invoked as follows:

```
certreq -dn dn -reqout filename -keyout filename
[-dsaparms filename] [-bits bits]
```

The arguments are:

`-reqout` *`filename`*

`certreq` writes a certificate request to this file. This file should be sent to a Certification Authority to be signed.

`-keyout` *`filename`*

`certreq` writes the private key to this file. Make sure that only the administrator has read permission for this file.

`-dn` *`dn`*

The distinguished name of the server or client.

`-dsaparms` *`filename`*

A Digital Signature Algorithm (DSA) key is based on a set of shared parameters. Two parameter sets are provided in the lib directory: `dsa512.pem` for 512 bit keys and `dsa1024.pem` for 1024 bit keys. This argument is not needed if an RSA key is to be generated.

`-bits` *`bits`*

The bits field should contain the number of bits of key to generate for the RSA algorithm, such as 512 or 1024.

For example, to create a RSA key in the file `k` and a certificate request in the file `r`:

```
certreq -reqout r -keyout k -dn "dc=iPlanet, dc=com" -bits 1024
```

To create a DSA key in the file `k` and the certificate request in the file `r`:

```
certreq -reqout r -keyout k -dn "dc=iPlanet, dc=com"
-dsaparms <server-root>/idar-<hostname>/etc/dsa512.pem
```

Key File

When RSA is being used, the file specified by the `-keyout` parameter will contain the RSA private key similar to the following:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDrYc78Q9PnU8Q5d0SfFyNXI84sRtGP9NXgP70XxY6Wdg3xoQAx
Z/xWyE/0dx0xRR8bzCpi25eTVyYbyQMxY6yu2OxvsywDYwkAN2bRBgePUMVSlx2j
Zi62Fr2CzsAaajbOO0yqXFP/gyjXphYorbOvyG78Xp3vFIesWYl6GgzglwIDAQAB
AoGBAOIm1E9N/+/XbMXl0Nmlyn+z2Ch0Vm6gx0kxFEYduvTXMmiAzwWpKipbRW7V
bwfiqJP1opfe8hBPgD7b8CRo5wQziQlypp7JnFnDjL7U/QARzRATUax/t8RdCxQ4
PrZ45At/amZpkkWCozYfXA+57LlhW535KxLcstMTNpNu7YPhAkEA+PgUO09eSzMs
YyAfvVlctmaLDmcapMLRlNp5oAnUYvwK17l0mCrATTgT3w1rQ1h2zKixlUSooqy5
W3fTN19ZMQJBAPIHgJvjGGKTvDVWM4kKOQgUm1KqaUgwBqfBnpSfA+kgJenSgRs9
aX1OM6xa0w+V3zwMSeF5VBLZ4X94kgsEZEcCQA/2iAWNfzQ/IbdxVdekJSekx4Gy
5qhtvVZX87hpKO73zhIq1+jxxMaus8d3ass0ntlcb5Zsgot7m57bvfUs7eECQC/s
nmG/wQdb+4uQKxo6pPpdojfnOHurztWO+EiziAG0dO1s2lW7flTqlD7PqTVP1uk8
AbEc5jHpZMZp6Hk4AGcCQQC2PtFQ4jykO/VdXETM3jUGPbyavxA7prdb8ut1fPfi
lzPtow1Lcn4uCK/sXecS+RaXEh3wPKJkaoTlUMJpivbO
-----END RSA PRIVATE KEY-----
```

This is the base 64 encoding of the RSA private key. The private key file must be protected against theft, tampering, or viewing.

Certificate Request File

The file specified by `-reqout` will contain a certificate request. This should be sent to the CA or discarded; iDAR does not make use of the certificate request. An example file is:

```
-----BEGIN CERTIFICATE REQUEST-----
MIIBbTCB1wIBADAuMRMwEQYKCZImiZPyLGQBGRYDY29tMRcwFQYKCZImiZPyLGQB
GRYHaVBsYW5ldDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA62HO/EPT51PE
OXdEnxcjVyPOLEbRj/TV4D+9F8WOlnYN8aEAMWf8VshP9HcdMUUfG8wqYtuXk1cm
G8kDMWOsrtjsb7MsA2MJADdm0QYHj1DFUpcdo2Yutha9gs7AGmo2zjtMqlxT/4Mo
16YWKK2zr8hu/F6d7xSHrFmJehoM4JcCAwEAAaAAMA0GCSqGSIb3DQEBBAUAA4GB
AHcBoLa3Bi3o+HblCIkD6Rx29gShLwVK+QyzrPHrC9iGgrOuZBcrJSqSVXx5U/iR
8ia14YKkZDqPJmEIe/eoxhKWgfxUQyi3D0DUovzg/+M0EIWxoCrwbJhl/1DvUgQl
eTLj+mPeW3cj4KSQdWa5TJxj6Zf2PvIMppVB0wmbkPfj
-----END CERTIFICATE REQUEST-----
```

The contents of the certificate request file are a base 64 encoding of a value of the ASN.1 type defined in PKS#10.

```
SIGNED { SEQUENCE {
version Version,
subject Name,
subjectPublicKeyInfo SubjectPublicKeyInfo,
attributes [0] IMPLICIT Attributes } }
```

Supported SASL Mechanisms

iDAR supports clients that bind to the directory using the SASL mechanism. However, not all SASL mechanisms are supported. Table 12-2 summarizes the current support.

Table 12-2 Supported SASL Mechanisms

CRAM-MD5	Yes
DIGEST-MD5 (auth only)	Yes
DIGEST-MD5 (connection protection)	No
EXTERNAL	No
GSSAPI	No

Chapter 13

Configuration Generation Tool

iPlanet Directory Access Router (iDAR) configuration generation tool is described in this chapter, in the following sections:

Introduction

iDAR's operation requires the creation of two independent sets of configuration information. The main iDAR configuration is large and can be unwieldy (in contrast to the startup configuration), and can be enumerated within a text file or it can be stored in an LDAP server. The advantage of storing the main configuration in an LDAP server is that it allows central administration of distributed set of iDAR deployments.

The startup configuration file possesses only the information necessary for iDAR to find the main configuration. Unlike the main configuration, which could be in an LDAP server, the startup configuration must be a text file on a file system accessible by the machine hosting the iDAR process.

Configuration Assistance

To simplify the generation of these two configuration files, iDAR is packaged with a command-line tool, `tailor`, that will generate working configuration files from a minimal set of user supplied parameters. The tool has been created to ease the chore of expressing the relation between iDAR, its clients, and its LDAP servers. However, the tool does not attempt to support some of iDAR's more complex configuration subtleties such as attribute renaming, subtree hiding, etc. Complex configuration such as those are best accomplished by using the tool as the starting point and then completing the tailoring process via iDAR's console-based administration facilities.

Configuration Tool's Parameters

The configuration tool `tailor` assumes it is being run on the environment that will host the iDAR that's being configured.

The tool has the following syntax:

```
tailor     [-startup outFile] [-ldif outFile] [-listen port]
           [-suffix dn] [-objectclass class]
           [-bind_dn dn -bind_pw password -url [ldap://]host[:port]]
           [-onbind [[ldap://]host[:port] [name] [secureport]]]
           [-server [ldap://]host[:port] [name] [secureport]]
           [-ssl_key file -ssl_cert file [-ssl_version number] [-ldaps port]]
           [-remember [outFile]] [-recall [file]]
```

The parameters follow a fixed pattern: a hyphenated key word / option followed by one or more values to associate with the option.

Tailor Options

The following recognized options are grouped by subject:

`-ldif outFile`

The `-ldif` option informs the configuration tool where to store the textual representation of the subsequently generated configuration file. The resulting text file can either be loaded directly by iDAR or imported into an LDAP server with certain provisos (i.e., the suffix is acceptable to the server and the server has supporting iDAR attributes and class configuration information).

The supplied `outFile` should be stipulated as a complete and absolute path to the destination file. An absolute path is required. The configuration tool will verify that you have sufficient permissions to create or replace the given file.

If this option is omitted, the tool will name the output file `tailor.ldif` and place it in the `etc` subdirectory subordinate to the installation's instance directory. No attempt is made by the tool to import the resulting configuration into an LDAP environment. That step requires end user manual intervention.

```
-bind_pw password
-bind_dn dn
-url [ldap://]host[:port]
```

These options are used in the case where iDAR is to retrieve its main configuration from an LDAP server. The `-bind_dn`, `-bind_pw`, and `-url` options are recorded into iDAR's startup configuration where they'll be used by iDAR when it contacts the configuration's host server.

NOTE All three of these options are required if any one of them is stipulated.

`-startup outFile`

This option instructs the tool where to place iDAR's startup configuration file.

Corollary to the `-ldif` option, the `outFile` option supplied here should also be a complete and absolute path to the destination file. Again, an absolute path is required. The configuration tool will verify that you have sufficient permissions to create or replace the given file.

If this option is omitted, the tool will name the output file `tailor.txt` and place it in the `etc` subdirectory subordinate to the installation's instance directory.

`-listen port`

This option provides the facility to dictate which TCP port number on which iDAR will listen for new connections. This option sets the value stipulated for `ids-proxy-con-listen-port` enumerated for the `ids-proxy-sch-GlobalConfiguration` objectclass entry.

The tool assumes a value of 389 for this option when it is not stipulated.

```
-onbind  [[ldap://]host[:port] [name] [secureport]]]
-server  [ldap://]host[:port] [name] [secureport]]
```

The `-server` option is used to enumerate the hosts that iDAR can contact; thus more than one `-server` could be specified. The optional `name` and `secureport` values are used to associate a name to the given host and to indicate which TCP port (if any) the host expects secured connections. Those two optional values may be stipulated in any order. In addition, either or both the `name` and the `secureport` directives may be omitted. The name given will be used to form part of the distinguished name in the properties section of the resulting configuration; thus, care should be exercised to avoid duplicating names. Note: the tool generates a default name in the case where one isn't given. Those default names begin with "`server-`" and are appended with a sequentially generated number to avoid duplication.

If the tool is given more than one `-server` option, the tool will emit a configuration that reflects load balancing directives. Note: iDAR assumes that a set of servers that are being load balanced are of equal content and of equal functionality. If the configuration tool isn't presented a `-server` option, it will act as if `-server ldap://localhost:389` has been stipulated.

The `-onbind` directive informs the configuration tool that it is to establish two distinct groups within the configuration: one group for anonymous binds and another for non-anonymous binds. The `-onbind` option's syntax is similar to `-server`'s syntax with the notable exception that it does not require the specification of a host name. If the host name is omitted, then `-onbind` will assume that the set of hosts to contact on successful bind is the same set as stipulated by the `-server` option(s). The key to recognize here is that connections that fall within `-onbind`'s purgative will still contact the same machines but the connections will have an expanded set of available operations. To the contrary, stipulating one or more `-onbind` hosts removes the possibility of supporting the rebinding of a SASL connection.

If you use `-onbind` to enumerate more than one host, then the configuration tool will set up additional properties to accommodate load balancing across the set of `-onbind` hosts.

```
-ssl_cert file
-ssl_key file
-ssl_version number
-ldaps port
```

These options enable iDAR to support encrypted connections from its clients. These options do not affect iDAR's ability to encrypt connections between itself and its corresponding servers. The `-ssl_cert` parameter specifies the absolute path to the file that possess the certificate that iDAR should employ. The `-ssl_key` option identifies the file that possesses the key iDAR should use in conjunction with the given certificate. These options map their values to the `ids-proxy-sch-GlobalConfiguration` objectclass' `ids-proxy-con-ssl-key` and `ids-proxy-con-ssl-cert` attributes.

If both `-ssl_key` and `-ssl_cert` are specified then the tool will also allow the specification of `-ssl_version` and / or `-ldaps`.

The `-ssl_version` must be one of the following values: 20, 23, 30, or 31. A value of 20 configures iDAR for SSL version 2 only, 30 selects SSL version 3 only, 23 permits either SSL version 2 or version 3 connections and 31 stipulates TLS version 1 only. This option maps its value to the `ids-proxy-sch-GlobalConfiguration` objectclass' `ids-proxy-con-ssl-version` attribute.

Setting a value for `-ldaps` instructs iDAR to listen on the given TCP port (in addition to the TCP port specified by the `-listen` directive) with the subtle difference being that only encrypted connections will be accepted on the `-ldaps` specified TCP port. This option maps its value to the `ids-proxy-sch-GlobalConfiguration` objectclass' `ids-proxy-con-ldaps-port` attribute.

`-suffix dn`

The tool will append the given suffix to all of the distinguished names used internally within the generated configuration. This option adds value to the resulting configuration if and only if the intent is to store the configuration in an LDAP server. Otherwise, the suffix just serves to clutter the distinguished names. The tool does assume that the given suffix already exists in the destined LDAP server.

`-objectclass class`

If your production deployment model calls for iDAR to keep its configuration in an LDAP server and you are employing schema checking, then you will probably need to supplement the configuration's entries with one or more objectclasses. By default, the configuration tool will tag each entry it creates with an objectclass of `ids-proxy-top`, but by stipulating this option (one or more times), you can specify the objectclass(es) to use instead.

`-remember outFile`
`-recall file`

The `-remember` and `-recall` options provide a mechanism to support the recording of the command-line parameters and their subsequent reuse. The `-remember` and `-recall` directives can be used simultaneously or independently of each other. When used together, the given file names may or may not be the same depending if you wish to overlay the given input (same names) or evolve the input (different names). The `-remember` and `-recall` options can be used in conjunction with any other options as the tool merely merges the current command line with the recalled values.

Configuration Examples

First, let's define the pseudo environment we have at our disposal.

- Servers:

```
ldap://red.iplanet.com:10389
ldap://blue.iplanet.com:389
ldap://pink.iplanet.com:10205
```

- iDAR:

```
ldap://bitter.iplanet.com:389
```

- Encrypted iDAR:

```
ldap://bitter.iplanet.com:636
```

Because `bitter.iplanet.com` is declared as supporting encryption, then the iDAR stationed at `bitter.iplanet.com` will support "Start TLS" operations over its non-secured port as well.

Let's further assume that DNS can resolve the aforementioned machines merely by their hostname, i.e., `red`, `blue`, `green`, etc. And the last assumption: the subsequent examples assume that the tool is being run from a UNIX environment.

Given these assumptions, we can now show you how the example configurations shown in Appendix A, "Sample Configuration Files" were generated.

The tool is capable of generating four different classes of configurations: straight through, load balancing, binding based operation filtering, and load balancing with binding based operation filtering.

Straight Through Configuration

In the straight through configuration, iDAR is not adding much value to the overall process; rather, it is just relaying the information from its clients to a server. The general expectation is that the straight through configuration would not have much application in a production deployment but it does have value when it comes to building confidence that your iDAR is inter-operating with your clients and your LDAP server. In this particular case, the command-line tool is given information regarding only which TCP port number the iDAR process should listen on and what single server should iDAR contact.

```
tailor -listen 389 -server ldap://pink.iplanet.com:10205
```

This will yield the first example configuration file found in Code Example A-1 on page 249. The output files were named `<server-root>/idar-<hostname>/etc/tailor.txt` and `<server-root>/idar-<hostname>/etc/tailor.ldif` as we didn't supply the `-startup` and `-ldif` options.

Load Balancing Configuration

This scenario expands upon the straight through configuration via the stipulation of more than one server to contact. The configuration tool will emit directives specifying that the given set of servers is to divide the connection load among themselves.

```
tailor -server pink:10205 -server red:10389 -server blue
```

Here the tool generates a configuration where `pink.iplanet.com` will receive 34 percent, `red.iplanet.com` receives 33 percent and `blue.iplant.com` receives 33 percent of the connections. The tool will apportion the set of servers equally based on the cardinality of the set stipulated. To provide 100 percent coverage, any remaining percentage is added to the first enumerated server.

Again you'll find the resulting configuration files as one of the examples in Appendix A, "Sample Configuration Files."

Binding Based Operation Filtering

Upon examination of the generated configuration files from the previous scenarios, you'll note that the configuration tool generated only one group. (Note: all connections through iDAR resolve to a group through which permissions, actions, and rules are derived.) In this scenario, the tool will generate two groups: one that is employed when a connection utilizes a non-anonymous bind and another group that acts as a catch-all.

The idea behind these two groups is to extend permissions to non-anonymous binding connections and to restrict the permissions to anonymous binding connections; specifically, anonymous clients will be given search capabilities only.

```
tailor -server blue -onbind
```

Load Balancing and Binding Based Operation Filtering

Creating a configuration that capitalizes on these two types of functions merely requires combining their parameter lists.

```
tailor -server pink:10205 -server red:10389 -server blue -onbind
```

The emitted configuration will instruct iDAR to load balance over the `pink`, `red`, and `blue` servers and anonymous binding connections will possess search privileges only.

See Appendix A for a listing of the resulting configuration file.

Startup Configuration File

The `tailor.txt` file contains the bootstrap information iDAR needs to locate its main configuration. The directives in this file dictate if iDAR will utilize an additional file for its main configuration or if iDAR will solicit its main configuration from an LDAP server. By default, iDAR expects to find the startup configuration file, `tailor.txt`, in the `etc` subdirectory of the installation's instance directory. Note: via the use of the command-line parameter `-t`, iDAR can be instructed to use an alternate file as its startup configuration file.

As an aid in supporting high availability configurations, a startup configuration file may list several contact points for the main configuration's retrieval. Contact points are delineated within the startup configuration file by the use of two keywords: `Begin` and `End`. iDAR will process the contact information one by one in the order given. iDAR's actions on each contact point depends on the type of the given contact point (either an LDAP URL or an absolute path name to a file).

For LDAP-URL-based contact points, iDAR will attempt to contact the given host. If the host is unwilling or unable to return a configuration, then iDAR will proceed to its next contact point (if any). If the host returns a configuration, then iDAR will edit the contents returned and will then either begin following the main configuration's directives or end its execution if the configuration was deemed invalid.

For file based contact points, iDAR will attempt to load the given file as its main configuration. If the specified configuration is missing or is deemed invalid, iDAR will end its execution. iDAR will not attempt to move to the next contact point once it encounters a file based contact point.

In the case where iDAR is retrieving its main configuration from an LDAP host, iDAR can bind to the host using one of three methods: anonymous, simple, or by using SASL.

Anonymous binding is accomplished by omitting the `configuration_bind_pw` and `configuration_bind_dn` directives. In other words, your startup configuration's contact information would specify a `configuration_url` directive and nothing else.

Simple binding is supported through the use of both the `configuration_bind_pw` and the `configuration_bind_dn` directives.

SASL binding requires the specification of the `sasl_bind_mechanism`, `conguration_bind_pw` and one (and only one) of the following directives: either `configuration_bind_dn` or `configuration_username`.

Startup Configuration's Keywords

Each enumerated contact point uses the keyword `Begin` to signify the start of a contact point entry. Conversely, each contact point entry is terminated by the keyword `End`. Every directive stipulated in a startup configuration file is expressed on a line by itself. Line continuation within the startup configuration is not recognized nor supported. The configuration's options are specified via an option, followed by a colon, and a value triplet.

configuration_url

The `configuration_url` option specifies either an LDAP directory server and the distinguished name of the entry in that directory where the iDAR configuration is stored, or a local file in LDIF format. For example, if the iDAR configuration is stored in an LDAP directory on host `ldap.iplanet.com` with the LDAP service running on port `389` and the distinguished name of the iDAR entry is "`ids-proxy-con-Server-Name=iDAR`", then the following should be added to the configuration file:

```
Begin
configuration_url:
ldap://ldap.iplanet.com:389/ids-proxy-con-Server-Name=iDAR
End
```

If the configuration is to be kept in an LDAP server, you would probably need to specify a suffix following the `ids-proxy-con-Server-Name=iDAR` in order to maintain compatibility with the host directory's naming context. For example:

```
Begin
configuration_url:
ldap://ldap.iplanet.com:389/ids-proxy-con-Server-Name=iDAR,
ou=services, dc=iPlanet, dc=com
End
```

Each startup configuration directive should be specified as one contiguous line within the configuration file.

NOTE Do not interpret the line wrapping in the `configuration_url` examples as an instruction to insert a line break into your configuration file.

In the case where the configuration is stored in a LDIF formatted file, i.e., `<server-root>/idar-<hostname>/etc/tailor.ldif`, the following should be added to the configuration file:

```
Begin
configuration_url:
file://<server-root>/idar-<hostname>/etc/tailor.ldif#ids-proxy-con-
Server-Name=iDAR
End
```

configuration_bind_dn

The configuration_bind_dn option specifies the distinguished name to use when iDAR binds to the LDAP server specified in the configuration_url option. If this option is not specified, and an LDAP directory is specified as the value of the configuration_url option, then iDAR binds anonymously to the directory. If it is specified, iDAR will perform a simple bind with this distinguished name and the value of configuration_bind_pw as the password. For example:

```
Begin
configuration_url:
ldap://ldap.iplanet.com:389/ids-proxy-con-Server-Name=iDAR
configuration_bind_dn: cn=Directory Manager
configuration_bind_pw: secret
End
```

The configuration_bind_dn option is not needed and ignored if configuration_url is of the “file” form. Note: the configuration_bind_dn and configuration_username directives are mutually exclusive.

configuration_bind_pw

The configuration_bind_pw option is used to specify the password to use when binding to the LDAP directory. The directive is used to specify the password to use for either simple or SASL based binding. In order to preserve security, the configuration file must be protected against unauthorized reading. The configuration_bind_pw option is not needed and ignored if configuration_url is of the “file” form. (See configuration_bind_dn for an example.)

configuration_username

The configuration_username option specifies the username to use when iDAR binds to the LDAP server specified in the configuration_url option. This option is used only if SASL bind mechanism is used. Note: the configuration_bind_dn and configuration_username directives are mutually exclusive.

```
Begin
configuration_url:
ldap://ldap.iplanet.com:389/ids-proxy-con-Server-Name=iDAR
configuration_username: administrator
configuration_bind_pw: secret
sasl_bind_mechanism: CRAM-MD5
End
```

sasl_bind_mechanism

The `sasl_bind_mechanism` option can be set to either `CRAM-MD5` or `DIGEST-MD5` depending on which SASL bind mechanism you want iDAR to use. iDAR will perform either a simple bind or an anonymous bind if this option is absent. `DIGEST-MD5` provides a higher level of security than `CRAM-MD5` but `DIGEST-MD5` has not been as widely adopted as `CRAM-MD5`.

Starting, Stopping, and Restarting iDAR

This chapter describes how to start, stop, and restart iPlanet Directory Access Router (iDAR) from the command line. To do the same from the console, see Chapter 4, “Starting, Restarting, and Stopping iDAR.”

This chapter contains the following sections:

Starting and Stopping iDAR

iDAR program runs as a UNIX daemon process or an NT service normally started during system boot time.

On all platforms, iDAR’s start program resides at:

`<server-root>/idar-<hostname>/start-idar`

The startup configuration file resides at:

`<server-root>/idar-<hostname>/etc/tailor.txt`

iDAR may be started and stopped via the scripts found at:

`<server-root>/idar-<hostname>`

The Windows NT Service Manager should be used to start and stop iDAR on Windows NT. On platforms other than Windows NT, iDAR will produce only a `core` image in case of a crash if its effective user ID is same as its real user ID. Therefore, if you want iDAR to produce a core, then you must set the `ids-proxy-con-userid` attribute in object class `ids-proxy-sch-GlobalConfiguration` to the same user who starts the iDAR process. By default, if iDAR is run by `root`, it changes its userid to `nobody`.

Supported Flags

The flags supported by the start and stop scripts are described in Table 14-1.

Table 14-1 Flags Supported By the Start and Stop Scripts

Flag	Description
`-d`	When this flag is present, iDAR will handle only a single incoming connection at a time, and will send more detailed internal tracing information to the log file. This flag should not be used during normal operation, because it will prevent the iDAR daemon from detaching from the controlling terminal.
`-D`	This flag tells iDAR to send more detailed tracing information to the log file. iDAR will still handle multiple client connections and run as a daemon. The `-d` and `-D` flags should be treated as mutually exclusive.
`-I`	This flag is applicable only on Windows NT, and is ignored on all other platforms. This flag must be used when starting iDAR program `ldapfwd` from the command line. The `-d` flag should be used in conjunction with the `-I` flag on Windows NT to get the debug output.
`-t <startup configuration file>`	This option can be used to specify an alternate startup configuration file.
`-s`	This option tells iDAR to send the initial log messages to the `syslogd` using the `LOG_DAEMON` facility. This flag is ignored on Windows NT. This is the default if the environment variable `IDAR_ROOT` is not defined.

Table 14-1 Flags Supported By the Start and Stop Scripts *(Continued)*

Flag	Description
`-M`	If this flag is specified, iDAR will spawn another process to monitor itself. In the case where iDAR exits ungracefully, the monitor process restarts iDAR after waiting for 30 seconds. This is not available on Windows NT.
`-v`	This flag prints the version information for iDAR. On Windows NT, this flag should be used from the command line only.

Restarting iDAR

On UNIX platforms, iDAR can be sent a `SIGHUP` signal to make it re-read its configuration. If the configuration is re-read successfully, iDAR will use this new configuration for future connections. Client connections that are already established will continue to use the old configuration until the clients disconnect.

To signal iDAR to re-read its configuration, use the `hup-idar` command found at `<server-root>/idar-<hostname>`.

Some attribute values cannot be changed using the `HUP` signal facility. For changes to the following configuration parameters, iDAR will have to be shut down and started again. These attributes include:

```
ids-proxy-con-listen-port
ids-proxy-con-listen-host
ids-proxy-con-ldaps-port
ids-proxy-con-foreground
ids-proxy-con-listen-backlog
ids-proxy-con-ssl-cert
ids-proxy-con-ssl-key
```

Also, the logging properties `ids-proxy-sch-LogProperty` cannot be changed using this facility.

On all platforms, a `restart-idar` command is found at `<server-root>/idar-<hostname>`. The restart command simply invokes the `stop-idar` and `start-idar` commands found in the aforementioned directory.

Chapter 15

Global Configuration

iPlanet Directory Access Router (iDAR) configuration can be represented as entries either in an LDAP directory or in a file in LDIF format. This chapter describes the schema for storing the iDAR global configuration information.

Throughout the remainder of this document, all configuration attributes are assumed to have a syntax of a case-insensitive UTF-8 string (`cis`) unless explicitly stated otherwise. Attribute names follow one of the following two conventions: `ids-proxy-sch-*` attributes are those that are part of iDAR configuration schema but do not have a direct effect on iDAR functionality; `ids-proxy-con-*` attributes directly affect iDAR functional behavior.

This chapter contains the following sections:

ids-proxy-sch-LDAPProxy Object Class

The `ids-proxy-sch-LDAPProxy` object contains information about the name of the iDAR server and the distinguished name (DN) of the `ids-proxy-sch-GlobalConfiguration` object. iDAR finds this object using the DN specified in the `configuration_url` attribute in the startup configuration file. This object contains the following mandatory attributes.

ids-proxy-sch-Global-Config-Dn

The `ids-proxy-sch-Global-Config-Dn` attribute specifies the distinguished name of the `ids-proxy-sch-Global-Configuration` object entry. This attribute must be present and has `dn` syntax.

ids-proxy-con-Server-Name

The `ids-proxy-con-Server-Name` attribute specifies the string name of the server. This attribute must be present, as it is used to form the Relative Distinguished Name (RDN) of entries of this class, as well as to locate group and property entries.

Locating Group, Property, Rule, and Action Objects

ids-proxy-sch-Group-Base, ids-proxy-sch-Property-Base, ids-proxy-sch-Rule-Base, ids-proxy-sch-Action-Base

These attributes specify the DN of the entries under which the group and property object entries reside. iDAR performs a one-level search with the respective attribute value as the base of the search and one of the following filters.

Object	LDAP Filter
group	`(&(objectclass=ids-proxy-sch-Group)(ids-proxy-sch-enable=TRUE)(ids-proxy-sch-belongs-to=<Server-Name>))`
property	`(&(objectclass=ids-proxy-sch-Property)(ids-proxy-sch-enable=TRUE)(ids-proxy-sch-belongs-to=<Server-Name>))`
rule	`(&(objectclass=ids-proxy-sch-Rule)(ids-proxy-sch-enable=TRUE)(ids-proxy-sch-belongs-to=<Server-Name>))`
action	`(&(objectclass=ids-proxy-sch-Action)(ids-proxy-sch-enable=TRUE)(ids-proxy-sch-belongs-to=<Server-Name>))`

`<Server-Name>` is the name of the server as specified in the `ids-proxy-con-Server-Name` attribute. The `ids-proxy-sch-Group-Base` attribute must be present in every entry of this class. The other three attributes are optional and may be omitted if no entries of that type are used.

ids-proxy-sch-GlobalConfiguration Object Class

The `ids-proxy-sch-GlobalConfiguration` object contains various global configuration attributes. Each configuration must have one entry of this object class.

ids-proxy-con-Config-Name

The `ids-proxy-con-config-Name` attribute specifies the string name of the configuration. This attribute must be present, as it is used to form the RDN of entries of this class.

ids-proxy-con-listen-port

The `ids-proxy-con-listen-port` attribute takes a value of the port number, 1 through 65535, on which to listen for incoming connections. The default, if this attribute is not present, is TCP port 389, as specified for LDAP. This port number must be different from that used by any other LDAP server running on the same host. On UNIX platforms the server must be started as root to listen on a port number below 1024.

For example, to have iDAR listen on port 10389, the following attribute/value pair should be in the `ids-proxy-sch-GlobalConfiguration` object entry:

```
ids-proxy-con-listen-port: 10389
```

ids-proxy-con-listen-host

The `ids-proxy-con-listen-host` attribute takes a value of the name or the IP address of the host interface on which the iDAR will listen for connections. This attribute is needed only if there are multiple network interfaces on the host running iDAR. If this attribute is not present or has the value "`localhost`", iDAR will listen on all available network interfaces.

ids-proxy-con-max-conns

The `ids-proxy-con-max-conns` attribute takes a value (greater than zero) of the maximum number of simultaneous client connections that iDAR will accept. For example, if you want to allow only ten simultaneous connections, the following attribute/value pair should be in the `ids-proxy-sch-GlobalConfiguration` object entry:

```
ids-proxy-con-max-conns:   10
```

To allow an unlimited number of simultaneous connections, do not specify this attribute.

ids-proxy-con-listen-backlog

The `ids-proxy-con-listen-backlog` attribute takes a value (greater than zero) of the maximum number of outstanding connections in the listening socket's queue. The default is `128` if the attribute is absent. This attribute should be present if more than 128 clients could attempt to contact iDAR simultaneously.

The maximum value depends on the underlying operating system configuration.

ids-proxy-con-ldaps-port

The `ids-proxy-con-ldaps-port` attribute takes as a value the port number on which to listen for `LDAPS` (LDAP over SSL) connections. By default, iDAR does not listen for connections from `LDAPS` clients. This attribute must be present to enable `LDAPS` connections from clients using this nonstandard function, with a value such as `636`. This value must be different from the `ids-proxy-con-listen-port` value. This option also requires TLS/SSL configuration (see Chapter 12, "Configuring Security").

ids-proxy-con-userid

This option specifies the userid under which iDAR will run. If iDAR was run as `root` then it will change its `uid` to the one specified here. The default is to switch to `nobody`. This option is not applicable on Windows NT.

ids-proxy-con-working-dir

This attribute specifies the directory from which iDAR should run. On startup, iDAR will change its working directory to the directory specified as value for this attribute. The default is `/tmp`. This attribute only takes effect on platforms other than Windows NT.

ids-proxy-con-include-logproperty

This attribute specifies the value of the `ids-proxy-con-name` attribute of the `ids-proxy-sch-LogProperty` property entry. This property should be defined by an entry if you want to customize logging. Refer to Chapter 20, “Logging” for details.

TLS/SSL Configuration Attributes

The `ids-proxy-con-ssl-cert` and `ids-proxy-con-ssl-key` attributes must be present in the `ids-proxy-sch-GlobalConfiguration` object entry if the Start TLS extended operation or `ids-proxy-con-ldaps-port` option is to be supported. The TLS policy of each group can be configured using the group attribute `ids-proxy-con-ssl-policy`.

ids-proxy-con-ssl-key

The `ids-proxy-con-ssl-key` attribute specifies the pathname of the file on disk containing the private key associated with the server’s certificate. Refer to the section “Generating a TLS Key Pair,” on page 182 for more details.

ids-proxy-con-ssl-cert

The `ids-proxy-con-ssl-cert` attribute specifies the path name of the file on disk containing the server’s own certificate signed by the CA. The certificate in the file must conform to the Public Key Cryptography Standards (PKCS#10) format. Refer to the section “Generating a TLS Key Pair,” on page 182 for more details.

ids-proxy-con-send-cert-as-client

This attribute has a boolean syntax and specifies whether iDAR will send its server certificate when making a TLS connection to an LDAP directory server. Set it to `TRUE` if you want iDAR to send its certificate to the backend LDAP directory server when making a TLS connection. The default is `FALSE`.

ids-proxy-con-server-ssl-version, ids-proxy-con-client-ssl-version

These options specify the version of SSL that iDAR will use. `ids-proxy-con-server-ssl-version` specifies the version of SSL it will use for connections between itself and backend LDAP servers while `ids-proxy-con-client-ssl-version` specifies the version between itself and the clients. Permitted values are `20` for SSL version 2 only, `30` for SSL version 3 only, `23` for SSL version 2 or SSL version 3, and `31` for TLS version 1 only. The default for both is `23`.

ids-proxy-con-ssl-cert-required

If the `ids-proxy-con-ssl-cert-required` attribute is set to `TRUE`, iDAR will require all clients that establish an SSL session to submit a certificate chain. iDAR will close the connection if a certificate chain is not submitted. Note that this option does not effect SSL sessions between iDAR and the backend servers. The default value for this attribute is `FALSE`.

ids-proxy-con-ssl-cafile

The `ids-proxy-con-ssl-cafile` attribute specifies the path of the file on disk containing trusted root certificates. These certificates must be in the PEM format.

Connection Pool Configuration Attributes

If the connection pool is enabled, iDAR will try to reuse existing connections to the backend LDAP servers. Switching on this option can give significant performance gain if the backend server is on a Wide Area Network (WAN).

ids-proxy-con-connection-pool

The `ids-proxy-con-connection-pool` attribute enables or disables the connection pool module. If this attribute is present with the value `TRUE` iDAR will preconnect to the directory servers. The default is `FALSE`.

ids-proxy-con-connection-pool-interval

The `ids-proxy-con-connection-pool-interval` attribute takes an integer value greater or equal to one. It specifies the interval in seconds at which iDAR will sample the incoming requests to anticipate future activity. The default is `15`.

ids-proxy-con-connection-pool-timeout

The `ids-proxy-con-connection-pool-timeout` attribute takes an integer value greater or equal to zero. It specifies the period of time in seconds after which an idle connection to an LDAP server will be terminated. If a zero value is specified, no timeout will be applied. The default is `30`. This value should be less than the idle connection timeout value of the backend LDAP server.

Debugging

This attribute is useful when debugging iDAR. It is available on UNIX platforms only.

ids-proxy-con-foreground

The `ids-proxy-con-foreground` attribute specifies whether the iDAR should remain in the foreground or detach from the controlling terminal. The default is `FALSE`, and the iDAR will detach. This attribute should not be specified in the `ids-proxy-sch-GlobalConfiguration` object entry except when debugging.

Chapter 16

Groups Configuration

iPlanet Directory Access Router (iDAR) groups describe how to identify an LDAP client, and the restrictions to enforce for clients that match that group. Clients are initially identified into a group based on the network address from which they connect. They may change their group after a successful bind. (See sections "ids-proxy-sch-OnBindSuccessRule Object Class," on page 236 and "ids-proxy-sch-ChangeGroupAction Object Class," on page 240 for more details.)

Groups are tested in the descending order of priority, specified by the `ids-proxy-con-priority` attribute. The order of evaluation of groups with equal priority is undefined. If no groups are found to match a client, the client's request will be rejected. There must be at least one group entry in the configuration specification.

This chapter contains the following sections:

ids-proxy-sch-Group Object Class

This object class contains the group object attributes.

ids-proxy-con-Name

The `ids-proxy-con-Name` attribute specifies the name of the group or property. This attribute value must be unique within the set of groups. This attribute must be present as it forms the RDN of entries of this class.

ids-proxy-con-Priority

The `ids-proxy-con-Priority` attribute specifies the order of evaluation. It takes an integer value greater than or equal to zero. If this attribute is not present its value is assumed to be 0. If the priority of two objects are the same, their order of evaluation is undefined. The object with the highest value is evaluated first. Thus, for example, the most generic group should have the lowest `ids-proxy-con-Priority` value. This is a single valued attribute.

ids-proxy-sch-Enable

The `ids-proxy-sch-Enable` attribute specifies if this object entry is part of the configuration or not. For an entry to be part of iDAR configuration, the `ids-proxy-sch-Enable` attribute must be present with the value `TRUE`. If the `ids-proxy-sch-Enable` attribute is not present or has the value `FALSE`, the object entry is ignored. This is a single valued attribute.

ids-proxy-sch-belongs-to

This is a multi-valued attribute that specifies which server configuration this object entry belongs to. The value of this attribute should be the value of the `ids-proxy-con-server-name` attribute in the `ids-proxy-sch-LDAPProxy` object entry. If it does not match, the entry is ignored.

ids-proxy-sch-NetworkGroup Object Class

This object class is sub-classed from the `ids-proxy-sch-Group` object class. Clients are identified to belong to this object class based on their IP address and/or domain name. Each of the `ids-proxy-sch-NetworkGroup` object entries must have at least one value for the `ids-proxy-con-Client` attribute.

ids-proxy-con-Client

The `ids-proxy-con-Client` attribute specifies the domain names or IP addresses of the hosts that match the `ids-proxy-sch-NetworkGroup` object entry. If a group is used only for clients to switch to when they bind, this attribute should be present with value `0.0.0.0`.

The value of the `ids-proxy-con-Client` attribute can be one of the following:

- A domain name, for example, `foo.iplanet.com`. Note that iDAR does not assume any domain suffix by default; therefore, complete domain names must be provided.
- A domain name suffix with a leading period, for example, `.iplanet.com`. All hosts with domain names that end in that suffix are considered to match.
- A single IP address in dotted decimal form, e.g., `198.214.11.1`.
- An IP network mask, in the form of `<network number>/<mask bits>`, for example, `198.241.11.0/24`. The first half is the network number and the second half indicates the number of bits of the network number necessary for matching.
- An IP network mask, in the form of a pair of dotted decimal quads, for example, `198.241.11.0/255.255.255.128`. The first half is a network number, the second half indicates the bits of the network number necessary for matching. For example, `198.214.11.0/255.255.255.128` will match a host with IP address `198.214.11.63` but not one with IP address `198.214.11.191`.
- If a group is only used for clients to switch to when they bind, this attribute should be present with value `0.0.0.0`.
- The string `ALL` (must be in all caps) matches all hosts.

Since the groups are checked in the order of priority, the most specific group should have the highest value. Generic groups, such as those with the attribute `ids-proxy-con-Client: ALL`, should have the lowest value, as no further groups will be checked.

NOTE Use of domain names or domain name suffixes requires `ids-proxy-con-reverse-dns-lookup` to be set to `TRUE`.

ids-proxy-con-include-property

The `ids-proxy-con-include-property` attribute is used to specify the property object entry to include in a group. The value of this attribute should be the value of the `ids-proxy-con-Name` attribute in the property object you want to include. This attribute is used to include more specialized configuration properties into a group.

For example, to include a property with `ids-proxy-con-Name:Load-Balance` you must specify the following:

```
ids-Proxy-con-include-property:Load-Balance
```

ids-proxy-con-include-rule

The `ids-proxy-con-include-rule` attribute is used to specify the rule object entry to include in a group. The value of this attribute should be the value of the `ids-proxy-con-Name` attribute in the rule object you want to include.

This attribute is used to include rules that can be evaluated upon completion of predetermined processes and actions that can be taken on the outcome of the evaluation.

For example, the `ids-proxy-sch-OnSuccessfulBindRule` is evaluated on the successful completion of a bind operation. If the rule evaluates to `true`, then a client may be able to change its group. For more information see Chapter 18, “Events Configuration.”

ids-proxy-con-ssl-policy

The `ids-proxy-con-ssl-policy` attribute can have one of the following three values: `ssl_required`, `ssl_optional`, or `ssl_unavailable`. If `ssl_optional` is specified, then a TLS / SSL connection will be established if the client requests TLS / SSL. If `ssl_required` is specified, then the client must establish a TLS / SSL session before performing any operation. iDAR will not permit a TLS / SSL session if `ssl_unavailable` is specified. This is the default.

For example, to make sure that all clients in a group establish a TLS/SSL session before performing any operation, specify the following:

```
ids-proxy-con-ssl-policy: ssl_required
```

For more information on setting up TLS, see Chapter 12, “Configuring Security.”

ids-proxy-con-tcp-no-delay

If the value of this option is set to TRUE, then iDAR will disable the Nagle Algorithm for connections between itself and *clients* that fall into this group. This option should only be set to FALSE if the network bandwidth between iDAR and clients is small. Setting this option to FALSE may create substantial performance degradation. The default is TRUE.

ids-proxy-con-allow-multi-ldapv2-bind

If this attribute is set to TRUE, iDAR will allow a LDAP v2 client to bind multiple times over a single session. The LDAP v2 specification (RFC 1777) does not allow a client to bind multiple times in one session. However, some clients expect this functionality, so this attribute can be used to interoperate with these clients. If you do set `ids-proxy-con-allow-multi-ldapv2-bind` to TRUE, you should set `ids-proxy-con-use-version` attribute in the `ids-proxy-sch-LDAPServer` objectclass to the value 3. This will prevent iDAR from sending any illegal operations to the backend server(s). The default value for this attribute is FALSE.

ids-proxy-con-reverse-dns-lookup

If `ids-proxy-con-reverse-dns-lookup` is set to FALSE, then iDAR will not perform a reverse DNS lookup to find the domain name of the connecting client. Setting this attribute to FALSE can sometimes significantly improve iDAR’s performance. If you have used a domain name or a domain name suffix as a value to `ids-proxy-con-client` attribute, you must not set `ids-proxy-con-reverse-dns-lookup` to FALSE, otherwise iDAR will not function properly. The default is TRUE.

ids-proxy-con-timeout

The `ids-proxy-con-timeout` attribute in an `ids-proxy-sch-NetworkGroup` defines a period of client inactivity, after which iDAR may close the connection to the client. The value is a number of seconds, typically 600 or more. The default, if this attribute is not present in an `ids-proxy-sch-NetworkGroup` object entry, is to not timeout connections. If TCP keepalives are not enabled, this attribute must be present to keep iDAR from being clogged by lost client connections.

Forwarding Binds

Once iDAR has accepted a connection from the client and matched a group, it will wait for the client to send the LDAP bind request. iDAR uses the `ids-proxy-con-bind-name`, `ids-proxy-con-permit-auth-none`, `ids-proxy-con-permit-auth-simple` and `ids-proxy-con-permit-auth-sasl` attributes to determine whether to pass the bind request to the server, or reject the bind request and close the client's connection.

If the client's bind passes these tests, iDAR will forward it to the server. If the server accepts the bind, the connection is established. If, however, the server returns an error indication for the bind request, iDAR will forward the error indication to the client, and then close the connection to the client, if the client was using LDAPv2.

ids-proxy-con-bind-name

The `ids-proxy-con-bind-name` attribute takes as a value a distinguished name. If a client provides a distinguished name in its bind that is not subordinate to a value in a `ids-proxy-con-bind-name` attribute, the bind will be rejected. This attribute must be present at least once in a group in order to allow any access. To allow all access for clients that match the `ids-proxy-con-client` attribute/value pair, regardless of bind name, this attribute should be present, with a zero-length value, as in

`ids-proxy-con-bind-name:`

This attribute has the `dn` syntax.

ids-proxy-con-permit-auth-none

The `ids-proxy-con-permit-auth-none` attribute specifies whether anonymous binds, when the client has not supplied a password, are permitted. The default is `TRUE`, if this attribute is not present in a group. This attribute may be changed to `FALSE` to forbid anonymous bind requests.

ids-proxy-con-permit-auth-simple

The `ids-proxy-con-permit-auth-simple` attribute specifies whether simple binds are permitted, when the client has supplied a password in the clear. The default is `TRUE`, if this attribute is not present in a group. This attribute may be changed to `FALSE` to forbid clear text password authenticated bind requests.

ids-proxy-con-permit-auth-sasl

The `ids-proxy-con-permit-auth-sasl` attribute specifies whether SASL binds are permitted. The default is `TRUE`, if this attribute is not present in a group. This attribute may be changed to `FALSE` to forbid SASL authentication.

Controlling Which Operations Are Forwarded

After a successful bind, iDAR by default forwards search and compare requests. iDAR also recognizes the unbind request, and closes the connection to the server.

ids-proxy-con-permit-op-search

If the `ids-proxy-con-permit-op-search` attribute is present in a group, with the value `FALSE`, iDAR will not forward search requests to the server. The default is `TRUE`.

ids-proxy-con-permit-op-compare

If the `ids-proxy-con-permit-op-compare` attribute is present in a group, with the value `FALSE`, iDAR will not forward compare requests to the server. The default is `TRUE`; however, few clients use this operation.

ids-proxy-con-permit-op-add, ids-proxy-con-permit-op-delete, ids-proxy-con-permit-op-modify, ids-proxy-con-permit-op-modrdn, ids-proxy-con-permit-op-extended

By default, iDAR does not forward Add, Modify, Delete, ModifyRDN, or Extended Operations requests. To permit forwarding of these operations, the attributes `ids-proxy-con-permit-op-add`, `ids-proxy-con-permit-op-delete`, `ids-proxy-con-permit-op-modify`, `ids-proxy-con-permit-op-modrdn`, and `ids-proxy-con-permit-op-extended` must be present in the `ids-proxy-sch-NetworkGroup` object entry, with the value `TRUE` for the appropriate operation to be allowed.

NOTE You must set `ids-proxy-con-permit-op-extended` to `TRUE` if you want your clients to be able to negotiate Start TLS.

Hiding a Subtree of Entries With ids-proxy-con-forbidden-subtree

The `ids-proxy-con-forbidden-subtree` attribute specifies the distinguished name of the base of a subtree of entries to be excluded. This is a multi-valued attribute. The default, if this attribute is not present in a group, is to allow access to all entries in the directory.

Operations that request entries at or below a forbidden subtree will be rejected with an insufficient access error. Entries that match a search filter and are inside a forbidden subtree are dropped. Note that this option does not remove DN syntax attributes whose values fall under the subtree from entries that are being returned as part of the result.

This attribute has dn syntax.

Attributes Controlling Search Requests

The `ids-proxy-con-filter-inequality` and `ids-proxy-con-min-substring-size` attributes are used to prevent certain kinds of searches from reaching the LDAP server. If the client's search request falls under either restriction, iDAR will return an insufficient access error to the client.

ids-proxy-con-filter-inequality

The `ids-proxy-con-filter-inequality` attribute specifies whether clients are permitted to request searches that contain inequality filters (`attr>=value`) and (`attr<=value`). The default is `TRUE` if this attribute is not present. This attribute should be present in the `ids-proxy-sch-NetworkGroup` object entry with the value `FALSE` if you do not wish to permit inequality searches to be performed.

ids-proxy-con-min-substring-size

The `ids-proxy-con-min-substring-size` attribute specifies the minimum permissible length of a substring in a search filter. The value is a number greater than one. The default, if this attribute is not present, is to allow any size of substring in a search filter. This attribute should be present in the `ids-proxy-sch-NetworkGroup` object entry if you wish to restrict the kinds of searches that may be performed by web robots, etc. For example, a value of 2 will block searches like`(cn=A*)`.

Controlling Search and Compare Requests

The `ids-proxy-con-forbidden-compare` and `ids-proxy-con-permitted-compare` attributes are used to prevent certain kinds of search and compare operations from reaching the LDAP server. If the client's request falls under this restriction, iDAR will return an insufficient access error to the client.

ids-proxy-con-forbidden-compare

The `ids-proxy-con-forbidden-compare` attribute specifies the name of an attribute that cannot be used by a client in a search filter or compare request. This is a multi-valued attribute. The default, if this attribute is not present in a group, and the `ids-proxy-con-permitted-compare` attribute is not present in the group either, is to permit all attributes to be used for search filters and comparisons.

For example, to allow clients to search on all attributes except `telephoneNumber` and `payrollCode`, add the following:

```
ids-proxy-con-forbidden-compare: telephoneNumber
ids-proxy-con-forbidden-compare: payrollCode
```

ids-proxy-con-permitted-compare

The `ids-proxy-con-permitted-compare` attribute specifies the name of an attribute that may be used in a search filter or compare request. This is a multi-valued attribute.

If there is one or more `ids-proxy-con-permitted-compare` attributes values present in an `ids-proxy-sch-NetworkGroup` object entry, and a compare does not match one of these, the request will be rejected by iDAR. If there are no `ids-proxy-con-permitted-compare` attributes present in a group, and an attribute does not match any `ids-proxy-con-forbidden-compare` attributes, then it may be used by clients.

For example, if you only want the `cn`, `dn`, and `mail` attributes to be searchable by the client, add the following to that `ids-proxy-sch-NetworkGroup` object entry:

```
ids-proxy-con-permitted-compare: cn
ids-proxy-con-permitted-compare: dn
ids-proxy-con-permitted-compare: mail
```

Attributes Modifying Search Requests

The `ids-proxy-con-minimum-base`, `ids-proxy-con-max-scope`, and `ids-proxy-con-max-timelimit` attributes may modify the search request before it is forwarded to the server.

ids-proxy-con-minimum-base

The `ids-proxy-con-minimum-base` attribute specifies the base of a subtree for all operations. This is a single valued attribute and has dn syntax. If this attribute is not present, then there is no restriction to a minimum base.

Operations whose target entry is at or below the minimum base entry are not affected by this attribute. If the target entry is superior to the minimum base entry, and the operation is a subtree search, then the query will be rewritten before being sent to the server, to change the target entry to be the minimum base. If the target entry is not below the minimum base or a superior of it, the request will be rejected with a no such object error.

For example, if the `ids-proxy-con-minimum-base` is set as:

```
ids-proxy-con-minimum-base: o=iPlanet, st=California, c=US
```

and a subtree search of `st=California, c=US` is received, the search will be rewritten such that the server performs a subtree search of `o=iPlanet, st=California, c=US`.

ids-proxy-con-max-scope

The `ids-proxy-con-max-scope` attribute specifies the maximum scope that a client may specify in a search request. This attribute may take one of the following values:

0: only `baseObject` searches may be requested
1: `baseObject` and `oneLevel` searches may be requested
2: `baseObject`, `oneLevel` and `subtree` may be requested

The default, if this attribute is not present, is to permit all search scopes.

ids-proxy-con-max-timelimit

The `ids-proxy-con-max-timelimit` attribute specifies a maximum time limit in seconds for search operations. If the client specifies a time limit that is larger than the value given in this attribute, this attribute will override the client's request. The default, if this attribute is not present in a group, is to allow the client to set any time limit, including no limit.

Attributes Restricting Search Responses

The `ids-proxy-con-max-result-size`, `ids-proxy-con-forbidden-return`, `ids-proxy-con-permitted-return`, and `ids-proxy-con-search-reference` attributes in a group describe restrictions that are applied to search results being returned by the server, before they are forwarded to the client.

ids-proxy-con-max-result-size

The `ids-proxy-con-max-result-size` attribute specifies the maximum number of result entries that may be returned to a client at one time from a single search operation. The value may be any number greater than zero, and if reached, will cause an `administrativeLimitExceeded` error to be indicated to the client and subsequent entries will be discarded. The default, if this attribute is not present, is to not discard entries.

ids-proxy-con-forbidden-return

The `ids-proxy-con-forbidden-return` attribute specifies the name of an attribute that is to be excluded from search result entries. This is a multi-valued attribute. The default, if this attribute is not present in the `ids-proxy-sch-NetworkGroup` object entry, and the `ids-proxy-con-permitted-return` attribute (discussed next) is not present in the group, is to permit all attributes to be returned.

For example, if you do not want the `homePhone` attribute to be returned in the search result, add the following attribute/value pair to the `ids-proxy-sch-NetworkGroup` object entry:

```
ids-proxy-con-forbidden-return: homePhone
```

ids-proxy-con-permitted-return

The `ids-proxy-con-permitted-return` attribute specifies the name of an attribute that may be returned from a search result, if present. This is a multi-valued attribute.

If there are `ids-proxy-con-permitted-return` attributes values present in an `ids-proxy-sch-NetworkGroup` object entry, and an attribute does not match one of those listed, it is not returned. If there are no `ids-proxy-con-permitted-return` attributes values present in an `ids-proxy-sch-NetworkGroup` object entry, and no value matches any `ids-proxy-con-forbidden-return` attribute values, then it is returned.

ids-proxy-con-search-reference

The `ids-proxy-con-search-reference` attribute specifies what to do if a search continuation reference is generated during a search. It can be one of the following values: `forward` (forward the reference to the client), `follow` (follow the reference and return the result to the client), or `discard`. The default is `discard`.

A search continuation referral is a special case of a referral whereby part of the query has been satisfied by the original directory server queried but that directory server has a reference to another directory server with more data satisfying the query.

This attribute can be used to hide the part of your Directory Information Tree whose naming context is mastered by another LDAP server. It also prevents clients from finding out the network address and port on which this server runs.

Controlling the Return of Referrals

ids-proxy-con-reference

Each group can have an attribute `ids-proxy-con-reference` that specifies whether to forward, follow, or discard referrals returned by the server. It takes the following values: `forward`, `follow`, or `discard`. The default is `forward`. It should be noted that a client that does not implement LDAPv3 will not understand forwarded referrals.

The use of this attribute is similar to the `ids-proxy-con-search-reference` attribute. This attribute applies to all referrals except for search continuation referrals.

ids-proxy-con-referral-ssl-policy

This attribute is similar to `ids-proxy-con-ssl-policy`. If `ssl_required` is specified, and an operation is referred, then a TLS/SSL session will be initiated before the operation is forwarded to the referred server. If `ssl_unavailable` is specified, TLS/SSL will not be initiated. If `ssl_optional` is specified, TLS/SSL will be initiated only if the client already has TLS/SSL session established with iDAR. The default, if this attribute is not present, is `ssl_unavailable`.

ids-proxy-con-referral-bind-policy

The `ids-proxy-con-referral-bind-policy` controls the bind policy when an operation is referred and the referral is being followed. The following three values can be specified: `bind_anonymous` (always bind anonymous), `bind_any` (use simple bind if the client had used password-based bind; or else bind as anonymous), or `bind_required` (reject the referred operation if the client is not password-based bound). The default is `bind_any`.

Note that iDAR cannot replay binds for clients bound using a SASL mechanism. Thus, the referral operation will be rejected if `bind_required` is specified and the client used a SASL mechanism to bind.

ids-proxy-con-max-refcount

The `ids-proxy-con-max-refcount` takes an integer value greater or equal to zero. It limits the maximum number of references that will be followed for a single operation. The default is 15. A value of zero indicates that no limit will be applied.

Controlling Server Load

The `ids-proxy-con-max-simultaneous-operations-per-connection` and `ids-proxy-con-max-operations-per-connection` attributes can be used to control the number of simultaneous operations and total number of operations a client can request on one connection. The `ids-proxy-con-max-conns` and `ids-proxy-con-max-simultaneous-conns-from-ip` can be used to restrict the number of connections for a group.

ids-proxy-con-max-simultaneous-operations-per-connection

The `ids-proxy-con-max-simultaneous-operations-per-connection` attribute limits the number of simultaneous operations iDAR will process per connection in that group. The value is an integer greater than zero. If this attribute is not present then no limit is enforced. For example, if you set

```
ids-proxy-con-max-simultaneous-operations-per-connection: 1
```

all the clients in that group will be forced to perform synchronous LDAP operations. Additional simultaneous requests, except for requests to abandon an operation, will fail with `Server Busy` error.

ids-proxy-con-max-operations-per-connection

The `ids-proxy-con-max-operations-per-connection` limits the total number of operations that iDAR will allow per connection in a group. The value is an integer greater than zero. If a client exceeds the maximum number of operations allowed for its group on one connection, that connection will be closed by iDAR. If this attribute is not present, then no limit is set.

ids-proxy-con-max-conns

This `ids-proxy-con-max-conns` attribute is the same attribute as the global attribute described in `ids-proxy-con-max-conns` on page 204. If specified within an `ids-proxy-sch-NetworkGroup`, then this attribute applies only to that group entry.

ids-proxy-con-max-simultaneous-conns-from-ip

The `ids-proxy-con-max-simultaneous-conns-from-ip` attribute can be used to restrict the number of simultaneous connections clients can make from a single IP address. The default is `0`, which means any number of connections are allowed.

Chapter 17

Properties Configuration

Property objects are used to identify more specialized restrictions that the clients must follow. These properties can then be included in other entries where the restrictions need to be applied.

This chapter contains the following sections:

ids-proxy-sch-Property Object Class

This object class defines the property object attributes. It contains the following attributes:

ids-proxy-con-Name

The `ids-proxy-con-Name` attribute specifies the name of the property. This attribute value must be unique within the set of properties. This attribute must be present as it forms the RDN of entries of this class.

ids-proxy-con-Priority

The `ids-proxy-con-Priority` attribute is reserved for future versions of this product.

ids-proxy-sch-Enable

The `ids-proxy-sch-Enable` attribute specifies if this object entry is part of the configuration or not. For an entry to be part of an iDAR configuration, the `ids-proxy-sch-Enable` attribute must be present with a value `TRUE`. If the `ids-proxy-sch-Enable` attribute is not present or has the value `FALSE`, the entry is ignored.

ids-proxy-sch-belongs-to

This is a multi-valued attribute that specifies which server configuration(s) this object entry belongs to. Each value of this attribute should be the value of the `ids-proxy-con-server-name` attribute in the `ids-proxy-sch-LDAPProxy` object entry, defined in the section “ids-proxy-con-Server-Name,” on page 202.

ids-proxy-sch-LoadBalanceProperty Object Class

This object class is sub-classed from the `ids-proxy-sch-property` object class and can be used to tell iDAR to load balance across multiple LDAP servers. Either this property can be included in a group using the `ids-proxy-con-include-property` attribute, or the group can have the `ids-proxy-con-server` attribute, but not both. The following attribute is mandatory for this object class.

Note that iDAR uses a monitor process to make health checks on its backend servers. This feature is automatically enabled if load balancing is used. iDAR makes an anonymous search operation for the Root DSE every 10 seconds for each of its backend directory servers. If one of them becomes unavailable or unresponsive, iDAR removes it from the active load balanced server set. When the server becomes available again, it is reintroduced in the set. In order for the monitoring feature to work efficiently, you must have configured the host on which iDAR is running according to the recommendations of the `<code&gtidsktune</code&gt` utility described in Chapter 2, “Computer System Requirements” of the *iDAR Installation Guide.*

ids-proxy-con-Server

The `ids-proxy-con-Server` attribute is used to specify the server name and the percentage of client connections to be served by it. Its value should have the following syntax:

`<server-Property>[#<percentage>]`

The `server-Property` is the value of `ids-proxy-con-Name` of an `ids-proxy-sch-LDAPServer` property object entry.

The `percentage` part is the percentage of total connections that this server should be sent.

The sum of all the `percentage` for the different servers specified must be 100. If the `percentage` is not specified, it is assumed to be zero. A server with zero percentage will be used as a failover server only.

For example, to load balance equally across three LDAP servers with `ids-proxy-con-name`, `ldap1`, `ldap2`, and `ldap3`, this attribute should have the following three values:

```
ids-proxy-con-Server: ldap1#33
ids-proxy-con-Server: ldap2#33
ids-proxy-con-Server: ldap3#34
```

ids-proxy-sch-SizeLimitProperty Object Class

This object class is sub-classed from the `ids-proxy-sch-property` object class. The `ids-proxy-sch-SizeLimitProperty` object class can be used to apply size limits based on the search base and search scope. If neither the search base nor search scope specified in this property object entry match a given search, the size limit defaults to the value of `ids-proxy-con-max-result-size` group attribute. This property class has the following three attributes.

ids-proxy-con-Size-Limit

The `ids-proxy-con-Size-Limit` attribute takes an integer value, specifying the size limit to enforce. This is a single valued attribute. The value of `-1` implies that no size limit will be enforced. This attribute is mandatory. Note that this is different from the LDAP protocol.

ids-proxy-con-Dn-One

If the DN of the search base of a one level search matches one of the distinguished names specified for the `ids-proxy-con-Dn-One` attribute, the value of `ids-proxy-con-Size-Limit` is enforced as the size limit of that search. This attribute has dn syntax.

ids-proxy-con-Dn-Sub

If the DN of the search base of a subtree search matches on the distinguished names specified for the `ids-proxy-con-Dn-Sub` attribute, the value of `ids-proxy-con-Size-Limit` is enforced as the size limit of that search. This attribute has dn syntax.

For example, to enforce a size limit of 50 for subtree searches with search base `dc=iplanet,dc=com` and one level searches with search base `ou=Development, dc=iplanet,dc=com`, the following attribute/value pairs should be placed in the `ids-proxy-sch-SizeLimitProperty` object entry:

```
ids-proxy-con-Size-Limit: 50
ids-proxy-con-Dn-One: dc=iPlanet, dc=com
ids-proxy-con-Dn-Sub: ou=Development, dc=iPlanet, dc=com
```

ids-proxy-sch-RenameAttributeProperty Object Class

This object class is sub-classed from the `ids-proxy-sch-property` object class. The `ids-proxy-sch-RenameAttributeProperty` object class can be used to rename attributes in client queries and in server results. This property has two mandatory single valued attributes.

ids-proxy-con-server-attr-name

The `ids-proxy-con-server-attr-name` attribute is used to specify the name of the attribute known to the server.

ids-proxy-con-client-attr-name

The `ids-proxy-con-client-attr-name` attribute is used to specify the name of the attribute known to the client.

If a client request contains an attribute name specified in `ids-proxy-con-client-attr-name`, it will be transformed to the value of `ids-proxy-con-server-attr-name`. Similarly, if a result sent by the server contains an attribute name specified in `ids-proxy-con-server-attr-name`, it will be transformed to the value of `ids-proxy-con-client-attr-name`.

ids-proxy-sch-ForbiddenEntryProperty Object Class

This object class is sub-classed from the `ids-proxy-sch-property` object class. The `ids-proxy-sch-ForbiddenEntryProperty` can be used to specify a list of entries or attributes in those entries that you want to hide. It has the following attributes.

ids-proxy-con-dn-exact

This is a multi-valued attribute that can be used to specify a list of DN's of entries or attributes in those entries that you want to hide.

ids-proxy-con-dn-regexp

This is a multi-valued attribute that can be used to specify a list of regular expressions of DN's of entries, or attributes in those entries that you want to hide. The regular expression of the DN should be specified in the normalized form, i.e., there should be no spaces between RDN components and the “=” sign, attribute names must be in all capital letters, and if the attribute values have the `cis` syntax, then they should also be capitalized.

For example, to match any DN with a RDN component of “`ou=internal`”, you must specify the following:

```
ids-proxy-con-dn-regexp: .*OU=INTERNAL.*
```

If there are `ids-proxy-con-permitted-return` attributes present in the `ids-proxy-sch-ForbiddenEntryProperty` object entry, and an attribute does not match one of those listed, it is not returned. If there are no `ids-proxy-con-permitted-return` attribute values are present in an `ids-proxy-sch-ForbiddenEntryProperty` object entry, and no attributes match any `ids-proxy-con-forbidden-return` attribute values, then it is returned.

The following book can be used as a reference on regular expressions: *Mastering Regular Expressions,* by Friedl and Oram, published by O'Reilly, ISBN: 1565922573.

ids-proxy-con-ava

The `ids-proxy-con-ava` is a multi-valued attribute that can be used to specify a list of attribute name / value pairs. If an entry has an attribute name / value pair that matches any of the specified attribute name / value, then that entry or some of its content will be hidden. The `ids-proxy-con-ava` attribute has the following syntax:

```
<attribute-name>:<attribute-value>
```

For example if you want to restrict all entries which have either "`ou=internal`" or "`secret=yes`" as one if its attributes, then you can specify the following:

```
ids-proxy-con-ava: ou:internal
ids-proxy-con-ava: secret:yes
```

ids-proxy-con-forbidden-return

The `ids-proxy-con-forbidden-return` attribute specifies the name of an attribute that is to be excluded from the entry that has matched any of the above specifications. This is a multi-valued attribute. If this attribute is not present and the `ids-proxy-con-permitted-return` attribute is also not present in the `ids-proxy-sch-ForbiddenEntryProperty`, then the entire entry is hidden.

ids-proxy-con-permitted-return

The `ids-proxy-con-permitted-return` attribute specifies the name of an attribute that may be returned as part of the entry that has matched any of the above specifications. This is a multi-valued attribute.

ids-proxy-sch-LDAPServer Object Class

This object class is sub-classed from the `ids-proxy-sch-property` object class. The `ids-proxy-sch-LDAPServer` property is used to define the backend LDAP servers. In order to reduce DNS lookup overhead, iDAR caches the DNS entry of the LDAP server host. Thus, if the IP address of your backend server changes, you must either restart iDAR or send it a HUP signal (UNIX only) so that it rereads its configuration.

ids-proxy-con-host

The `ids-proxy-con-host` attribute specifies the full DNS or IP address name of the host where the backend LDAP server is running. This attribute is mandatory.

ids-proxy-con-port

The `ids-proxy-con-port` attribute specifies the port number on which the backend LDAP server is running. The default port used if this attribute is absent is `389`.

ids-proxy-con-sport

The `ids-proxy-con-sport` attribute takes as a value the port number on which the backend server listens for `LDAPS` (LDAP over SSL) connections. Note that this is the non standard mechanism for supporting LDAP over SSL. Do not set any value for this attribute if the backend LDAP server does not support `LDAPS`.

ids-proxy-con-supported-version

The `ids-proxy-con-supported-version` specifies if the server defined by this entry supports only LDAP version 2 (`2`) or both LDAP version 2 and LDAP version 3 (`23`). The default is `23`.

ids-proxy-con-use-version

This attribute tells iDAR the preferred LDAP protocol version to use when talking to the backend server this entry defines. Permitted values are:

2	Only use LDAPv2
3	Only use LDAPv3
23	Use the version that the client is using. This is also the default.

This option is useful when you have an LDAPv2 client for which iDAR needs to follow referrals. In this case iDAR itself needs to connect as LDAPv3 client to the backend server in order for the backend server to send referrals back to it. This attribute value should be set to 3 if the attribute `ids-proxy-con-allow-multi-ldapv2-bind` in object class `ids-proxy-sch-GlobalConfiguration` is set to `TRUE`.

ids-proxy-con-tcp-no-delay

If the `ids-proxy-con-tcp-no-delay` attribute is set to `FALSE`, then iDAR will enable the Nagel Algorithm on connections to this *server*. This option must be set to `FALSE` only if the network bandwidth between iDAR and the server defined by this object entry is very limited. The default is `TRUE`.

ids-proxy-con-link-security-policy

The `ids-proxy-con-link-security-policy` defines the security policy for connections between iDAR and the backend server. If it is set to `ssl_always`, iDAR will establish a TLS session with the backend server before performing any operation. If it is set to `ssl_never`, then iDAR will not establish a TLS session with the backend server. Finally, if `ssl_optional` is specified, iDAR will only establish a TLS session with the backend server when the client establishes one with iDAR. This is the default behavior.

ids-proxy-con-x509cert-subject

The `ids-proxy-con-x509cert-subject` specifies the name of the LDAP server as specified by the `subject` field of its certificate. This attribute has a `dn` syntax. If this attribute is specified, iDAR will match it with the value of `subject` attribute present in the server certificate. iDAR will reject a TLS session with the LDAP server in case of a mismatch.

This attribute allows iDAR to authenticate the LDAP server it is connecting to.

ids-proxy-con-keepalive-interval

The value of `ids-proxy-con-keepalive-interval` is used by iDAR to determine if the network link to an LDAP directory server is down or if the LDAP directory server has become unresponsive. If the client connected to iDAR has pending operations and if iDAR has not received any data from the connection's LDAP server for the number of seconds specified by `ids-proxy-con-keepalive-interval` attribute, then iDAR will test the availability of the LDAP server by opening another communication channel to it. If iDAR is unsuccessful in doing so, it will fail over to another LDAP server if available. If it is successful in establishing a second channel, iDAR will wait for another interval to receive data from the LDAP server. If iDAR has still not received any data from the LDAP directory server, it will assume that the LDAP server is in an unresponsive state and will fail over to another directory server if available. The default value for this attribute is 180 seconds. It is recommended that you increase this value if the LDAP server is not on the same local network as iDAR.

ids-proxy-sch-LogProperty Object Class

This object class is sub-classed from the `ids-proxy-sch-property` object class. The `ids-proxy-sch-LogProperty` object class can be used to define the logging characteristics of iDAR. This property can only be referred by the `ids-proxy-sch-GlobalConfiguration` object entry, using the `ids-proxy-con-include-logproperty` attribute.

Please refer to the section "ids-proxy-sch-LogProperty Object Class," on page 243 for more details of this property.

Chapter 18

Events Configuration

Event objects are used to specify conditions that are evaluated by iPlanet Directory Access Router (iDAR) at predetermined states. If these conditions are satisfied, then iDAR could optionally take certain actions as defined in the chapter on Actions.

This chapter contains the following sections:

ids-proxy-sch-Rule Object Class

This object class defines the event object attributes. It contains the following attributes.

ids-proxy-con-Name

The `ids-proxy-con-Name` attribute specifies the name of the event. This attribute value must be unique within the set of events. This attribute must be present as it forms the RDN of entries of this class.

ids-proxy-con-Priority

The `ids-proxy-con-Priority` attribute specifies the order of evaluation, and is reserved for future versions of this product.

ids-proxy-sch-Enable

The `ids-proxy-sch-Enable` attribute specifies if this object entry is part of the configuration or not. For an entry to be part of iDAR configuration, the `ids-proxy-sch-Enable` attribute must be present with a value `TRUE`. If the `ids-proxy-sch-Enable` attribute is not present or has the value `FALSE`, the entry is ignored.

ids-proxy-sch-belongs-to

This is a multi-valued attribute that specifies which server configuration(s) this object entry belongs to. Each value of this attribute should be the value of the `ids-proxy-con-server-name` attribute in the `ids-proxy-sch-LDAPProxy` object entry.

ids-proxy-con-execute

The `ids-proxy-con-execute` is a multi-valued attribute that specifies the actions to take if the event evaluates to `TRUE`. If this action does not exist then no actions will be taken. The `ids-proxy-con-execute` attribute has the following syntax:

`<action-name>#<precedence>`

The `action-name` part is the value of the `ids-proxy-con-Name` attribute in the `ids-proxy-sch-Action` object entry that is to be executed.

The `precedence` part is a positive integer. This integer determines the order of execution of the actions. Actions with higher precedence are executed first. If two actions have the same precedence, their order of execution is undefined.

ids-proxy-sch-OnBindSuccessRule Object Class

This object class is sub-classed from the `ids-proxy-sch-Rule` object class. This event is evaluated when a client successfully completes a bind operation.

ids-proxy-con-ssl-required

If the `ids-proxy-con-ssl-required` attribute is set to `TRUE` then the `ids-proxy-sch-OnBindSuccessrule` event will evaluate to true only if the client has established an SSL session with iDAR. The default is `FALSE`.

ids-proxy-con-bind-anonymous

If the `ids-proxy-con-bind-anonymous` attribute is set to `TRUE` then the `ids-proxy-sch-OnBindSuccessrule` event will evaluate to true when the condition in `ids-proxy-con-ssl-required` is satisfied and the client has just completed a successful anonymous bind.

ids-proxy-con-bind-simple

If the `ids-proxy-con-bind-simple` attribute is set to `TRUE` then the `ids-proxy-sch-OnBindSuccessrule` event will evaluate to true when the condition in `ids-proxy-con-ssl-required` is satisfied and the client has just completed a successful password-based bind.

ids-proxy-con-bind-sasl

If the `ids-proxy-con-bind-sasl` attribute is set to `TRUE` then the `ids-proxy-sch-OnBindSuccessrule` event will evaluate to true when the condition in `ids-proxy-con-ssl-required` is satisfied and the client has just completed a successful bind using any SASL mechanism.

ids-proxy-sch-OnSSLEstablishedRule Object Class

This object class is sub-classed from the `ids-proxy-sch-Rule` object class. This event is evaluated when a client successfully establishes an SSL session. This event does not have any attributes.

Chapter 19

Actions Configuration

Action objects are used to specify actions to take if an event evaluates to `TRUE`. This chapter contains the following sections:

- ids-proxy-sch-Action Object Class (page 239)
- ids-proxy-sch-ChangeGroupAction Object Class (page 240)

ids-proxy-sch-Action Object Class

This object class defines the action object attributes. It contains the following attributes.

ids-proxy-con-Name

The `ids-proxy-con-Name` attribute specifies the name of the action. This attribute value must be unique within the set of actions. This attribute must be present as it forms the RDN of entries of this class.

ids-proxy-con-Priority

The `ids-proxy-con-Priority` attribute is reserved for future versions of this product.

ids-proxy-sch-Enable

The `ids-proxy-sch-Enable` attribute specifies if this object entry is part of the configuration or not. For an entry to be part of the iDAR configuration, the `ids-proxy-sch-Enable` attribute must be present with a value `TRUE`. If the `ids-proxy-sch-Enable` attribute is not present or has the value `FALSE`, the entry is ignored.

ids-proxy-sch-belongs-to

This is a multi-valued attribute that specifies which server configuration(s) this object entry belongs to. Each value of this attribute should be the value of the `ids-proxy-con-server-Name` attribute in the `ids-proxy-sch-LDAPProxy` object entry.

ids-proxy-sch-ChangeGroupAction Object Class

This object class is sub-classed from the `ids-proxy-sch-Action` object class. It can be used to specify the groups to which a client can change. It has the following multi-valued attribute.

ids-proxy-con-to-group

The `ids-proxy-con-to-group` specifies the group to which the client can change. It has the following syntax:

```
( <group-name>|NONE )

#<normal-DN-regexp>#<precedence>
```

`group-name`: The value of `ids-proxy-con-Name` attribute from a `ids-proxy-sch-Group` object entry. The special value `NONE` indicates no change is to be made.

`normal-DN-regexp`: The clients DN must match this regular expression. This regular expression must be normalized, i.e., there should be no spaces in between RDN components and “=” sign, all attribute names must be capitalized, and attribute values with `cis` syntax must be capitalized.

`precedence`: This must be a positive integer. This determines the order in which the change of group will be tried. `ids-proxy-con-to-group` value with higher value will be tried first.

For example, if you want all clients who have `cn=manager` as their least significant RDN component in their bind DN, into the manager group and not do anything with other bind credentials, you must specify the following:

```
ids-proxy-con-to-group: manager#CN=MANAGER,.*#100
ids-proxy-con-to-group: NONE#.*#10
```

The following book can be used as a reference on regular expressions: *Mastering Regular Expressions,* by Friedl and Oram, published by O'Reilly, ISBN: 1565922573.

Chapter 20

Logging

Logging is configured in iPlanet Directory Access Router (iDAR) by the `ids-proxy-sch-LogProperty` object class.

ids-proxy-sch-LogProperty Object Class

This object class is sub-classed from the `ids-proxy-sch-property` object class. The `ids-proxy-sch-LogProperty` object class can be used to define the logging characteristics of iDAR. This property can only be referred by the `ids-proxy-sch-GlobalConfiguration` object entry, using the `ids-proxy-con-include-logproperty` attribute.

Before iDAR reads this property, log messages are either written to a file on disk or, optionally, iDAR can send them to the `syslog` daemon, on platforms other than Windows NT. You cannot send the log messages to both a file and to the `syslog` daemon at the same time.

If the environment variable `IDAR_ROOT` is defined, then logs are written to the file `"IDAR_ROOT"/logs/fwd.log`. The `"IDAR_ROOT"/logs` directory must exist with write permissions to iDAR.

On platforms other than Windows NT, you can optionally send these initial log messages to the `syslog` daemon by specifying the `-s` flag on startup. iDAR uses the `daemon` facility, with the `warning`, `info` and `debug` priorities. If you choose this route, make sure that your `syslogd` is properly configured. For example, to have all the messages written to a particular file `/var/adm/messages`, the following line must be added to the file `/etc/syslog.conf`:

```
daemon.warning;daemon.info;daemon.debug  /var/adm/messages
```

If the `ids-proxy-sch-LogProperty` object entry is not specified in a configuration, then iDAR continues to log in the default location. This object class has the following attributes.

ids-proxy-con-log-level

The `ids-proxy-con-log-level` attribute specifies the amount of logging iDAR will do. It can have any one of the following values in order of detail.

```
critical
exception
warning
notice
trace
detail_trace
```

A more detailed log level automatically implies all previous levels. For example, if the value of `warning` is specified, which is also the default, then `exception` and `critical` level messages will also be logged.

ids-proxy-con-stat-level

The `ids-proxy-con-stat-level` attribute specifies the different kinds of statistics iDAR will log. The value of this attribute is a `'|'` separated list of the following options.

Table 20-1 Logging Levels

Option	Description
none	No statistics will be logged.
mods	Statistics about operations that write to the directory like add, modify, and delete will be logged.
op	Statistics about all LDAP operations will be logged.
conn	Statistics about network connections will be logged.
stat	General statistics such as how many clients are connected will be logged.
audit	Audit information such as DN of client who just completed a bind / unbind will be logged.
full	All of the above except none.

For example, if you want to log audit and connection statistics only, you must specify the following in the `ids-proxy-sch-LogProperty` object entry.

```
ids-proxy-con-stat-level: conn|audit
```

The default is none.

ids-proxy-con-log-syslog

The `ids-proxy-con-log-syslog` attribute specifies the syslog facility code if logging is being done through the `syslogd` daemon. The attribute is not available on Windows NT. The default is `LOG_DAEMON`. On all platforms other than Windows NT, and if the environment variable `IDAR_ROOT` is not defined, logging is done throughout the `syslogd` daemon by default. If `IDAR_ROOT` is defined, logging is done to the file `$(IDAR_ROOT)/logs/fwd.log` by default. See Chapter 14, “Starting, Stopping, and Restarting iDAR” for more information.

ids-proxy-con-log-file

The `ids-proxy-con-log-file` takes as value the full path name of the file you want the log messages to go to. Only one of the `ids-proxy-con-log-syslog` and `ids-proxy-con-log-file` attributes must be specified. On Windows NT, if this attribute is not specified then the log messages are sent to `"%IDAR_ROOT%\logs\fwd.log"` by default.

ids-proxy-con-audit-syslog

The `ids-proxy-con-audit-syslog` attribute is essentially the same as the `ids-proxy-con-log-syslog` attribute. However, this attribute only impacts the audit log messages. If this attribute is absent, all audit messages are logged with other log messages specified by the `ids-proxy-con-log-syslog` or `ids-proxy-con-log-file` attributes.

ids-proxy-con-audit-file

The `ids-proxy-con-audit-file` attribute is essentially the same as the `ids-proxy-con-log-file` attribute. However, this attribute only impacts the audit log messages. Only one of `ids-proxy-con-audit-syslog` or `ids-proxy-con-audit-file` attributes can be specified. If this attribute is absent, all audit messages are logged with other log messages specified by the `ids-proxy-con-log-syslog` or `ids-proxy-con-log-file` attributes.

Part 4

Appendixes

Appendix A, “Sample Configuration Files”

Appendix B, “iDAR Error Guide”

Appendix C, “iDAR FAQ, Features, and Troubleshooting”

Appendix A

Sample Configuration Files

iPlanet Directory Access Router (iDAR) provides a configuration tool named `tailor` that is capable of generating the `tailor.ldif` files for four different classes of configurations: straight through, load balancing, binding based operation filtering, and load balancing with binding based operation filtering. This appendix contains example configurations of all four types that were generated by the `tailor` configuration tool.

The appendix has the following sections:

Straight Through Configuration

Code Example A-1 shows a sample LDIF file generated by the `tailor` configuration tool in the “straight through configuration” mode.

Code Example A-1 Straight Through Configuration’s LDIF

```
version: 1

dn: ou=actions
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: actions
```

Code Example A-1 Straight Through Configuration's LDIF *(Continued)*

```
dn: ou=global
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: global

dn: ou=groups
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: groups

dn: ou=properties
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: properties

dn: ou=rules
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: rules

#
# This bootstrap entry defines:
#
# 1) the name of the iDAR server set is: "iDAR"
# 2) the DN of the ids-proxy-sch-GlobalConfiguration object
# 3) the base DN of the Properties definitions
# 4) the base DN of the Groups definitions
#
# Note: iDAR learns this entry's DN via the startup configuration (tailor.txt).
#
dn: ids-proxy-con-Server-Name=iDAR
objectclass: ids-proxy-sch-LDAPProxy
objectclass: ids-proxy-top
ids-proxy-con-Server-Name: iDAR
ids-proxy-sch-Global-Config-Dn: ids-proxy-con-Config-Name=config,ou=global
ids-proxy-sch-Group-Base: ou=groups
ids-proxy-sch-Property-Base: ou=properties
ids-proxy-sch-Rule-Base: ou=rules

#
# This Global properties entry declares:
#
# 1) which port the proxy will listen on for connections
#
dn: ids-proxy-con-Config-Name=config,ou=global
objectclass: ids-proxy-sch-GlobalConfiguration
objectclass: ids-proxy-top
ids-proxy-con-Config-Name: config
ids-proxy-con-connection-pool: FALSE
ids-proxy-con-connection-pool-interval: 15
ids-proxy-con-connection-pool-timeout: 30
ids-proxy-con-include-logproperty: LogProperty
ids-proxy-con-listen-backlog: 128
ids-proxy-con-listen-host: localhost
```

Code Example A-1 Straight Through Configuration's LDIF *(Continued)*

```
ids-proxy-con-listen-port: 389
ids-proxy-con-one-thread-only: FALSE
ids-proxy-con-userid: nobody
ids-proxy-con-working-dir: /tmp
ou: global

#
# Here we stipulate to iDAR where to send its log file as well
# as what logging level to use.
#
dn: ids-proxy-con-Name=LogProperty,ou=properties
objectclass: ids-proxy-sch-LogProperty
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: LogProperty
ids-proxy-con-log-file: /opt/iDAR/logs/fwd.log
ids-proxy-con-log-level: warning
ids-proxy-con-stat-level: none
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# iDAR finds its group properties by essentially performing a one level search
# beneath the ids-proxy-sch-LDAPProxy stipulated ids-proxy-sch-Group-Base.
# (This is in contrast with all other entries which are "found" by an exact DN
# to utilize.)  Since there could be more than one entry beneath the
# ids-proxy-sch-Group-Base, each group entry has an ids-proxy-con-Priority
# value to indicate their evaluation order. In addition, the
# ids-proxy-sch-belongs-to attribute allows entries at this level to exclude
# or include themselves based on if their ids-proxy-sch-belongs-to value
# matches the value asserted by the ids-proxy-con-Server-Name value found
# in the ids-proxy-sch-LDAPProxy objectclass entry.
#
# The following entry defines a group called "default" that:
#
# 1) assigns an evaluation priority of 33 (evaluates before 32, 31, 30, etc.)
# 2) declares the server configuration this entry belongs to as "iDAR"
# 3) allows all clients to be a member of the group as con-Client is ALL
# 4) allows all clients regardless of bind name (since it isn't enumerated)
# 5) allows all operations
# 6) stipulates that referrals are returned to the proxy's client
# 7) stipulates that the backend server for this group is described by the
#    ids-proxy-sch-LDAPServer objectclass entry named server-1.
#
dn: ids-proxy-con-Name=default,ou=groups
objectclass: ids-proxy-sch-Group
objectclass: ids-proxy-sch-NetworkGroup
objectclass: ids-proxy-top
ids-proxy-con-Client: ALL
ids-proxy-con-Name: default
ids-proxy-con-Priority: 33
ids-proxy-con-Server: server-1
ids-proxy-con-allow-multi-ldapv2-bind: FALSE
```

Code Example A-1 Straight Through Configuration's LDIF *(Continued)*

```
ids-proxy-con-filter-inequality: TRUE
ids-proxy-con-max-refcount: 15
ids-proxy-con-max-scope: 2
ids-proxy-con-max-simultaneous-conns-from-ip: 0
ids-proxy-con-permit-auth-none: TRUE
ids-proxy-con-permit-auth-sasl: TRUE
ids-proxy-con-permit-auth-simple: TRUE
ids-proxy-con-permit-op-add: TRUE
ids-proxy-con-permit-op-compare: TRUE
ids-proxy-con-permit-op-delete: TRUE
ids-proxy-con-permit-op-extended: TRUE
ids-proxy-con-permit-op-modify: TRUE
ids-proxy-con-permit-op-modrdn: TRUE
ids-proxy-con-permit-op-search: TRUE
ids-proxy-con-reference: forward
ids-proxy-con-referral-bind-policy: bind_any
ids-proxy-con-referral-ssl-policy: ssl_unavailable
ids-proxy-con-search-reference: follow
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-con-timeout: 120
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: groups

#
# server enumeration
#
dn: ids-proxy-con-Name=server-1,ou=properties
objectclass: ids-proxy-sch-LDAPServer
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: server-1
ids-proxy-con-host: pink.iplanet.com
ids-proxy-con-link-security-policy: ssl_optional
ids-proxy-con-port: 10205
ids-proxy-con-supported-version: 23
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties
```

Load Balancing Configuration

Code Example A-2 shows a sample LDIF file generated by the `tailor` configuration tool in the "load balancing configuration" mode.

Code Example A-2 Load Balancing Configuration's LDIF

```
version: 1

dn: ou=actions
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: actions

dn: ou=global
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: global

dn: ou=groups
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: groups

dn: ou=properties
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: properties

dn: ou=rules
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: rules

#
# This bootstrap entry defines:
#
# 1) the name of the iDAR server set is: "iDAR"
# 2) the DN of the ids-proxy-sch-GlobalConfiguration object
# 3) the base DN of the Properties definitions
# 4) the base DN of the Groups definitions
#
# Note: iDAR learns this entry's DN via the startup configuration (tailor.txt).
#
dn: ids-proxy-con-Server-Name=iDAR
objectclass: ids-proxy-sch-LDAPProxy
objectclass: ids-proxy-top
ids-proxy-con-Server-Name: iDAR
ids-proxy-sch-Global-Config-Dn: ids-proxy-con-Config-Name=config,ou=global
ids-proxy-sch-Group-Base: ou=groups
ids-proxy-sch-Property-Base: ou=properties
ids-proxy-sch-Rule-Base: ou=rules

#
# This Global properties entry declares:
#
# 1) which port the proxy will listen on for connections
#
dn: ids-proxy-con-Config-Name=config,ou=global
objectclass: ids-proxy-sch-GlobalConfiguration
```

Code Example A-2 Load Balancing Configuration's LDIF *(Continued)*

```
objectclass: ids-proxy-top
ids-proxy-con-Config-Name: config
ids-proxy-con-connection-pool: FALSE
ids-proxy-con-connection-pool-interval: 15
ids-proxy-con-connection-pool-timeout: 30
ids-proxy-con-include-logproperty: LogProperty
ids-proxy-con-listen-backlog: 128
ids-proxy-con-listen-host: localhost
ids-proxy-con-listen-port: 389
ids-proxy-con-one-thread-only: FALSE
ids-proxy-con-userid: nobody
ids-proxy-con-working-dir: /tmp
ou: global

#
# Here we stipulate to iDAR where to send its log file as well
# as what logging level to use.
#
dn: ids-proxy-con-Name=LogProperty,ou=properties
objectclass: ids-proxy-sch-LogProperty
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: LogProperty
ids-proxy-con-log-file: /opt/iDAR/logs/fwd.log
ids-proxy-con-log-level: warning
ids-proxy-con-stat-level: none
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# iDAR finds its group properties by essentially performing a one level search
# beneath the ids-proxy-sch-LDAPProxy stipulated ids-proxy-sch-Group-Base.
# (This is in contrast with all other entries which are "found" by an exact DN
# to utilize.)  Since there could be more than one entry beneath the
# ids-proxy-sch-Group-Base, each group entry has an ids-proxy-con-Priority
# value to indicate their evaluation order. In addition, the
# ids-proxy-sch-belongs-to attribute allows entries at this level to exclude
# or include themselves based on if their ids-proxy-sch-belongs-to value
# matches the value asserted by the ids-proxy-con-Server-Name value found
# in the ids-proxy-sch-LDAPProxy objectclass entry.
#
# The following entry defines a group called "default" that:
#
# 1) assigns an evaluation priority of 33 (evaluates before 32, 31, 30, etc.)
# 2) declares the server configuration this entry belongs to as "iDAR"
# 3) allows all clients to be a member of the group as con-Client is ALL
# 4) allows all clients regardless of bind name (since it isn't enumerated)
# 5) allows all operations
# 6) stipulates that referrals are returned to the proxy's client
# 7) stipulates that this group is to use load balancing.
#
dn: ids-proxy-con-Name=default,ou=groups
objectclass: ids-proxy-sch-Group
```

Code Example A-2 Load Balancing Configuration's LDIF *(Continued)*

```
objectclass: ids-proxy-sch-NetworkGroup
objectclass: ids-proxy-top
ids-proxy-con-Client: ALL
ids-proxy-con-Name: default
ids-proxy-con-Priority: 33
ids-proxy-con-allow-multi-ldapv2-bind: FALSE
ids-proxy-con-filter-inequality: TRUE
ids-proxy-con-include-property: load-balance
ids-proxy-con-max-refcount: 15
ids-proxy-con-max-scope: 2
ids-proxy-con-max-simultaneous-conns-from-ip: 0
ids-proxy-con-permit-auth-none: TRUE
ids-proxy-con-permit-auth-sasl: TRUE
ids-proxy-con-permit-auth-simple: TRUE
ids-proxy-con-permit-op-add: TRUE
ids-proxy-con-permit-op-compare: TRUE
ids-proxy-con-permit-op-delete: TRUE
ids-proxy-con-permit-op-extended: TRUE
ids-proxy-con-permit-op-modify: TRUE
ids-proxy-con-permit-op-modrdn: TRUE
ids-proxy-con-permit-op-search: TRUE
ids-proxy-con-reference: forward
ids-proxy-con-referral-bind-policy: bind_any
ids-proxy-con-referral-ssl-policy: ssl_unavailable
ids-proxy-con-search-reference: follow
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-con-timeout: 120
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: groups

#
# The following entry defines the property used to stipulate which servers
# will be contacted. It also apportions the servers.
#
# Note: it is assumed that the load balanced servers are equal in content
# and in capability.
#
dn: ids-proxy-con-Name=load-balance,ou=properties
objectclass: ids-proxy-sch-LoadBalanceProperty
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: load-balance
ids-proxy-con-Server: server-1#34
ids-proxy-con-Server: server-2#33
ids-proxy-con-Server: server-3#33
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# server enumeration
#
dn: ids-proxy-con-Name=server-1,ou=properties
```

Code Example A-2 Load Balancing Configuration's LDIF *(Continued)*

```
objectclass: ids-proxy-sch-LDAPServer
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: server-1
ids-proxy-con-host: pink
ids-proxy-con-link-security-policy: ssl_optional
ids-proxy-con-port: 10205
ids-proxy-con-supported-version: 23
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# server enumeration
#
dn: ids-proxy-con-Name=server-2,ou=properties
objectclass: ids-proxy-sch-LDAPServer
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: server-2
ids-proxy-con-host: red
ids-proxy-con-link-security-policy: ssl_optional
ids-proxy-con-port: 10389
ids-proxy-con-supported-version: 23
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# server enumeration
#
dn: ids-proxy-con-Name=server-3,ou=properties
objectclass: ids-proxy-sch-LDAPServer
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: server-3
ids-proxy-con-host: blue
ids-proxy-con-link-security-policy: ssl_optional
ids-proxy-con-port: 389
ids-proxy-con-supported-version: 23
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties
```

Binding Based Operation Filtering Configuration

Code Example A-3 shows a sample LDIF file generated by the `tailor` configuration tool in the “binding based operation filtering configuration” mode.

Code Example A-3 Binding Based Operation Filtering’s LDIF

```
version: 1

dn: ou=actions
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: actions

dn: ou=global
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: global

dn: ou=groups
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: groups

dn: ou=properties
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: properties

dn: ou=rules
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: rules

#
# This bootstrap entry defines:
#
# 1) the name of the iDAR server set is: "iDAR"
# 2) the DN of the ids-proxy-sch-GlobalConfiguration object
# 3) the base DN of the Properties definitions
# 4) the base DN of the Groups definitions
#
# Note: iDAR learns this entry's DN via the startup configuration (tailor.txt).
#
dn: ids-proxy-con-Server-Name=iDAR
objectclass: ids-proxy-sch-LDAPProxy
objectclass: ids-proxy-top
ids-proxy-con-Server-Name: iDAR
ids-proxy-sch-Action-Base: ou=actions
ids-proxy-sch-Global-Config-Dn: ids-proxy-con-Config-Name=config,ou=global
ids-proxy-sch-Group-Base: ou=groups
ids-proxy-sch-Property-Base: ou=properties
ids-proxy-sch-Rule-Base: ou=rules
```

Code Example A-3 Binding Based Operation Filtering's LDIF *(Continued)*

```
#
# This Global properties entry declares:
#
# 1) which port the proxy will listen on for connections
#
dn: ids-proxy-con-Config-Name=config,ou=global
objectclass: ids-proxy-sch-GlobalConfiguration
objectclass: ids-proxy-top
ids-proxy-con-Config-Name: config
ids-proxy-con-connection-pool: FALSE
ids-proxy-con-connection-pool-interval: 15
ids-proxy-con-connection-pool-timeout: 30
ids-proxy-con-include-logproperty: LogProperty
ids-proxy-con-listen-backlog: 128
ids-proxy-con-listen-host: localhost
ids-proxy-con-listen-port: 389
ids-proxy-con-one-thread-only: FALSE
ids-proxy-con-userid: nobody
ids-proxy-con-working-dir: /tmp
ou: global

#
# Here we stipulate to iDAR where to send its log file as well
# as what logging level to use.
#
dn: ids-proxy-con-Name=LogProperty,ou=properties
objectclass: ids-proxy-sch-LogProperty
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: LogProperty
ids-proxy-con-log-file: /opt/iDAR/logs/fwd.log
ids-proxy-con-log-level: warning
ids-proxy-con-stat-level: none
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# note that we change group on bind
#
dn: ids-proxy-con-Name=successfulBind,ou=rules
objectclass: ids-proxy-sch-OnBindSuccessRule
objectclass: ids-proxy-sch-Rule
objectclass: ids-proxy-top
ids-proxy-con-Name: successfulBind
ids-proxy-con-bind-anonymous: FALSE
ids-proxy-con-bind-sasl: TRUE
ids-proxy-con-bind-simple: TRUE
ids-proxy-con-execute: change#100
ids-proxy-con-ssl-required: FALSE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: rules
```

Code Example A-3 Binding Based Operation Filtering's LDIF *(Continued)*

```
#
# this is the group we changeto on successful bind
#
dn: ids-proxy-con-Name=change,ou=actions
objectclass: ids-proxy-sch-Action
objectclass: ids-proxy-sch-ChangeGroupAction
objectclass: ids-proxy-top
ids-proxy-con-Name: change
ids-proxy-con-to-group: onbind#.*#5
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: actions

#
# iDAR finds its group properties by essentially performing a one level search
# beneath the ids-proxy-sch-LDAPProxy stipulated ids-proxy-sch-Group-Base.
# (This is in contrast with all other entries which are "found" by an exact DN
# to utilize.)  Since there could be more than one entry beneath the
# ids-proxy-sch-Group-Base, each group entry has an ids-proxy-con-Priority
# value to indicate their evaluation order. In addition, the
# ids-proxy-sch-belongs-to attribute allows entries at this level to exclude
# or include themselves based on if their ids-proxy-sch-belongs-to value
# matches the value asserted by the ids-proxy-con-Server-Name value found
# in the ids-proxy-sch-LDAPProxy objectclass entry.
#
# The following entry defines a group called "onbind" that:
#
# 1) assigns an evaluation priority of 10 (evaluates before 9, 8, 7, etc.)
# 2) declares the server configuration this entry belongs to as "iDAR"
# 3) uses a con-Client value that FAILS matching during priority scanning
# 4) allows all clients regardless of bind name (since it isn't enumerated)
# 5) allows all operations
# 6) stipulates that referrals are returned to the proxy's client
# 7) stipulates that the backend server for this group is described by the
#    ids-proxy-sch-LDAPServer objectclass entry named server-1.
#
dn: ids-proxy-con-Name=onbind,ou=groups
objectclass: ids-proxy-sch-Group
objectclass: ids-proxy-sch-NetworkGroup
objectclass: ids-proxy-top
ids-proxy-con-Client: 0.0.0.0
ids-proxy-con-Name: onbind
ids-proxy-con-Priority: 10
ids-proxy-con-Server: server-1
ids-proxy-con-allow-multi-ldapv2-bind: FALSE
ids-proxy-con-filter-inequality: TRUE
ids-proxy-con-max-refcount: 15
ids-proxy-con-max-scope: 2
ids-proxy-con-max-simultaneous-conns-from-ip: 0
ids-proxy-con-permit-auth-none: TRUE
ids-proxy-con-permit-auth-sasl: TRUE
ids-proxy-con-permit-auth-simple: TRUE
ids-proxy-con-permit-op-add: TRUE
```

Code Example A-3 Binding Based Operation Filtering's LDIF *(Continued)*

```
ids-proxy-con-permit-op-compare: TRUE
ids-proxy-con-permit-op-delete: TRUE
ids-proxy-con-permit-op-extended: TRUE
ids-proxy-con-permit-op-modify: TRUE
ids-proxy-con-permit-op-modrdn: TRUE
ids-proxy-con-permit-op-search: TRUE
ids-proxy-con-reference: forward
ids-proxy-con-referral-bind-policy: bind_any
ids-proxy-con-referral-ssl-policy: ssl_unavailable
ids-proxy-con-search-reference: follow
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-con-timeout: 120
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: groups

#
# The following entry defines a group called "default" that:
#
# 1) assigns an evaluation priority of 33 (evaluates before 32, 31, 30, etc.)
# 2) declares the server configuration this entry belongs to as "iDAR"
# 3) allows all clients to be a member of the group as con-Client is ALL
# 4) allows all clients regardless of bind name (since it isn't enumerated)
# 5) disallows add, compare, delete, modify, modrdn and extended operations
# 6) stipulates that referrals are returned to the proxy's client
# 7) stipulates that the backend server for this group is described by the
#    ids-proxy-sch-LDAPServer objectclass entry named server-1.
#
dn: ids-proxy-con-Name=default,ou=groups
objectclass: ids-proxy-sch-Group
objectclass: ids-proxy-sch-NetworkGroup
objectclass: ids-proxy-top
ids-proxy-con-Client: ALL
ids-proxy-con-Name: default
ids-proxy-con-Priority: 33
ids-proxy-con-Server: server-1
ids-proxy-con-allow-multi-ldapv2-bind: FALSE
ids-proxy-con-filter-inequality: TRUE
ids-proxy-con-include-rule: successfulBind
ids-proxy-con-max-refcount: 15
ids-proxy-con-max-scope: 2
ids-proxy-con-max-simultaneous-conns-from-ip: 0
ids-proxy-con-permit-auth-none: TRUE
ids-proxy-con-permit-auth-sasl: TRUE
ids-proxy-con-permit-auth-simple: TRUE
ids-proxy-con-permit-op-add: FALSE
ids-proxy-con-permit-op-compare: FALSE
ids-proxy-con-permit-op-delete: FALSE
ids-proxy-con-permit-op-extended: FALSE
ids-proxy-con-permit-op-modify: FALSE
ids-proxy-con-permit-op-modrdn: FALSE
ids-proxy-con-permit-op-search: TRUE
ids-proxy-con-reference: forward
ids-proxy-con-referral-bind-policy: bind_any
```

Code Example A-3 Binding Based Operation Filtering's LDIF *(Continued)*

```
ids-proxy-con-referral-ssl-policy: ssl_unavailable
ids-proxy-con-search-reference: follow
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-con-timeout: 120
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: groups

#
# server enumeration
#
dn: ids-proxy-con-Name=server-1,ou=properties
objectclass: ids-proxy-sch-LDAPServer
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: server-1
ids-proxy-con-host: blue
ids-proxy-con-link-security-policy: ssl_optional
ids-proxy-con-port: 389
ids-proxy-con-supported-version: 23
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties
```

Load Balancing With Binding Based Operation Filtering Configuration

Code Example A-4 shows a sample LDIF file generated by the `tailor` configuration tool in the "load balancing with binding based operation filtering configuration" mode.

Code Example A-4 Load Balancing and Binding Based Filtering's LDIF

```
version: 1

dn: ou=actions
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: actions

dn: ou=global
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: global
```

Code Example A-4 Load Balancing and Binding Based Filtering's LDIF *(Continued)*

```
dn: ou=groups
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: groups

dn: ou=properties
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: properties

dn: ou=rules
objectclass: ids-proxy-top
objectclass: organizationalUnit
ou: rules

#
# This bootstrap entry defines:
#
# 1) the name of the iDAR server set is: "iDAR"
# 2) the DN of the ids-proxy-sch-GlobalConfiguration object
# 3) the base DN of the Properties definitions
# 4) the base DN of the Groups definitions
#
# Note: iDAR learns this entry's DN via the startup configuration (tailor.txt).
#
dn: ids-proxy-con-Server-Name=iDAR
objectclass: ids-proxy-sch-LDAPProxy
objectclass: ids-proxy-top
ids-proxy-con-Server-Name: iDAR
ids-proxy-sch-Action-Base: ou=actions
ids-proxy-sch-Global-Config-Dn: ids-proxy-con-Config-Name=config,ou=global
ids-proxy-sch-Group-Base: ou=groups
ids-proxy-sch-Property-Base: ou=properties
ids-proxy-sch-Rule-Base: ou=rules

#
# This Global properties entry declares:
#
# 1) which port the proxy will listen on for connections
#
dn: ids-proxy-con-Config-Name=config,ou=global
objectclass: ids-proxy-sch-GlobalConfiguration
objectclass: ids-proxy-top
ids-proxy-con-Config-Name: config
ids-proxy-con-connection-pool: FALSE
ids-proxy-con-connection-pool-interval: 15
ids-proxy-con-connection-pool-timeout: 30
ids-proxy-con-include-logproperty: LogProperty
ids-proxy-con-listen-backlog: 128
ids-proxy-con-listen-host: localhost
ids-proxy-con-listen-port: 389
ids-proxy-con-one-thread-only: FALSE
ids-proxy-con-userid: nobody
ids-proxy-con-working-dir: /tmp
```

Code Example A-4 Load Balancing and Binding Based Filtering's LDIF *(Continued)*

```
ou: global

#
# Here we stipulate to iDAR where to send its log file as well
# as what logging level to use.
#
dn: ids-proxy-con-Name=LogProperty,ou=properties
objectclass: ids-proxy-sch-LogProperty
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: LogProperty
ids-proxy-con-log-file: /opt/iDAR/logs/fwd.log
ids-proxy-con-log-level: warning
ids-proxy-con-stat-level: none
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# note that we change group on bind
#
dn: ids-proxy-con-Name=successfulBind,ou=rules
objectclass: ids-proxy-sch-OnBindSuccessRule
objectclass: ids-proxy-sch-Rule
objectclass: ids-proxy-top
ids-proxy-con-Name: successfulBind
ids-proxy-con-bind-anonymous: FALSE
ids-proxy-con-bind-sasl: TRUE
ids-proxy-con-bind-simple: TRUE
ids-proxy-con-execute: change#100
ids-proxy-con-ssl-required: FALSE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: rules

#
# this is the group we changeto on successful bind
#
dn: ids-proxy-con-Name=change,ou=actions
objectclass: ids-proxy-sch-Action
objectclass: ids-proxy-sch-ChangeGroupAction
objectclass: ids-proxy-top
ids-proxy-con-Name: change
ids-proxy-con-to-group: onbind#.*#5
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: actions

#
# iDAR finds its group properties by essentially performing a one level search
# beneath the ids-proxy-sch-LDAPProxy stipulated ids-proxy-sch-Group-Base.
# (This is in contrast with all other entries which are "found" by an exact DN
# to utilize.)  Since there could be more than one entry beneath the
# ids-proxy-sch-Group-Base, each group entry has an ids-proxy-con-Priority
```

Code Example A-4 Load Balancing and Binding Based Filtering's LDIF *(Continued)*

```
# value to indicate their evaluation order. In addition, the
# ids-proxy-sch-belongs-to attribute allows entries at this level to exclude
# or include themselves based on if their ids-proxy-sch-belongs-to value
# matches the value asserted by the ids-proxy-con-Server-Name value found
# in the ids-proxy-sch-LDAPProxy objectclass entry.
#
# The following entry defines a group called "onbind" that:
#
# 1) assigns an evaluation priority of 10 (evaluates before 9, 8, 7, etc.)
# 2) declares the server configuration this entry belongs to as "iDAR"
# 3) uses a con-Client value that FAILS matching during priority scanning
# 4) allows all clients regardless of bind name (since it isn't enumerated)
# 5) allows all operations
# 6) stipulates that referrals are returned to the proxy's client
# 7) stipulates that this group is to use load balancing.
#
dn: ids-proxy-con-Name=onbind,ou=groups
objectclass: ids-proxy-sch-Group
objectclass: ids-proxy-sch-NetworkGroup
objectclass: ids-proxy-top
ids-proxy-con-Client: 0.0.0.0
ids-proxy-con-Name: onbind
ids-proxy-con-Priority: 10
ids-proxy-con-allow-multi-ldapv2-bind: FALSE
ids-proxy-con-filter-inequality: TRUE
ids-proxy-con-include-property: load-balance
ids-proxy-con-max-refcount: 15
ids-proxy-con-max-scope: 2
ids-proxy-con-max-simultaneous-conns-from-ip: 0
ids-proxy-con-permit-auth-none: TRUE
ids-proxy-con-permit-auth-sasl: TRUE
ids-proxy-con-permit-auth-simple: TRUE
ids-proxy-con-permit-op-add: TRUE
ids-proxy-con-permit-op-compare: TRUE
ids-proxy-con-permit-op-delete: TRUE
ids-proxy-con-permit-op-extended: TRUE
ids-proxy-con-permit-op-modify: TRUE
ids-proxy-con-permit-op-modrdn: TRUE
ids-proxy-con-permit-op-search: TRUE
ids-proxy-con-reference: forward
ids-proxy-con-referral-bind-policy: bind_any
ids-proxy-con-referral-ssl-policy: ssl_unavailable
ids-proxy-con-search-reference: follow
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-con-timeout: 120
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: groups

#
# The following entry defines a group called "default" that:
#
# 1) assigns an evaluation priority of 33 (evaluates before 32, 31, 30, etc.)
# 2) declares the server configuration this entry belongs to as "iDAR"
```

Code Example A-4 Load Balancing and Binding Based Filtering's LDIF *(Continued)*

```
# 3) allows all clients to be a member of the group as con-Client is ALL
# 4) allows all clients regardless of bind name (since it isn't enumerated)
# 5) disallows add, compare, delete, modify, modrdn and extended operations
# 6) stipulates that referrals are returned to the proxy's client
# 7) stipulates that this group is to use load balancing.
#
dn: ids-proxy-con-Name=default,ou=groups
objectclass: ids-proxy-sch-Group
objectclass: ids-proxy-sch-NetworkGroup
objectclass: ids-proxy-top
ids-proxy-con-Client: ALL
ids-proxy-con-Name: default
ids-proxy-con-Priority: 33
ids-proxy-con-allow-multi-ldapv2-bind: FALSE
ids-proxy-con-filter-inequality: TRUE
ids-proxy-con-include-property: load-balance
ids-proxy-con-include-rule: successfulBind
ids-proxy-con-max-refcount: 15
ids-proxy-con-max-scope: 2
ids-proxy-con-max-simultaneous-conns-from-ip: 0
ids-proxy-con-permit-auth-none: TRUE
ids-proxy-con-permit-auth-sasl: TRUE
ids-proxy-con-permit-auth-simple: TRUE
ids-proxy-con-permit-op-add: FALSE
ids-proxy-con-permit-op-compare: FALSE
ids-proxy-con-permit-op-delete: FALSE
ids-proxy-con-permit-op-extended: FALSE
ids-proxy-con-permit-op-modify: FALSE
ids-proxy-con-permit-op-modrdn: FALSE
ids-proxy-con-permit-op-search: TRUE
ids-proxy-con-reference: forward
ids-proxy-con-referral-bind-policy: bind_any
ids-proxy-con-referral-ssl-policy: ssl_unavailable
ids-proxy-con-search-reference: follow
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-con-timeout: 120
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: groups

#
# The following entry defines the property used to stipulate which servers
# will be contacted. It also apportions the servers.
#
# Note: it is assumed that the load balanced servers are equal in content
# and in capability.
#
dn: ids-proxy-con-Name=load-balance,ou=properties
objectclass: ids-proxy-sch-LoadBalanceProperty
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: load-balance
ids-proxy-con-Server: server-1#34
ids-proxy-con-Server: server-2#33
```

Code Example A-4 Load Balancing and Binding Based Filtering's LDIF *(Continued)*

```
ids-proxy-con-Server: server-3#33
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# server enumeration
#
dn: ids-proxy-con-Name=server-1,ou=properties
objectclass: ids-proxy-sch-LDAPServer
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: server-1
ids-proxy-con-host: pink
ids-proxy-con-link-security-policy: ssl_optional
ids-proxy-con-port: 10205
ids-proxy-con-supported-version: 23
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# server enumeration
#
dn: ids-proxy-con-Name=server-2,ou=properties
objectclass: ids-proxy-sch-LDAPServer
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: server-2
ids-proxy-con-host: red
ids-proxy-con-link-security-policy: ssl_optional
ids-proxy-con-port: 10389
ids-proxy-con-supported-version: 23
ids-proxy-con-tcp-no-delay: TRUE
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties

#
# server enumeration
#
dn: ids-proxy-con-Name=server-3,ou=properties
objectclass: ids-proxy-sch-LDAPServer
objectclass: ids-proxy-sch-Property
objectclass: ids-proxy-top
ids-proxy-con-Name: server-3
ids-proxy-con-host: blue
ids-proxy-con-link-security-policy: ssl_optional
ids-proxy-con-port: 389
ids-proxy-con-supported-version: 23
ids-proxy-con-tcp-no-delay: TRUE
```

Code Example A-4 Load Balancing and Binding Based Filtering's LDIF *(Continued)*

```
ids-proxy-sch-Enable: TRUE
ids-proxy-sch-belongs-to: iDAR
ou: properties
```

Appendix B

iDAR Error Guide

The error numbers and text are logged by iPlanet Directory Access Router (iDAR) if some exception condition occurs. Only those errors that warrant an explanation are listed in this appendix.

The appendix has the following sections:

Unnumbered Errors

These errors are unnumbered and will happen only if an error occurs before iDAR could initialize itself.

`Registry settings are not present`

Installation problem or the Registry entries deleted by the Administrator.

`Could not open log file %s`

On Windows NT this means `%IDAR_ROOT%\logs\fwd.log` is not accessible where `IDAR_ROOT` is the variable in the registry.

On UNIX platforms, iDAR was not started with `-s` option and `${IDAR_ROOT}/logs/fwd.log` is not accessible.

`RegisterServiceCtrlHandler call failed`
`StartServiceCtrlDispatcher call failed`

Only on Windows NT. Administrator did not reboot after installation. Or, bad installation. Or, Administrator tried to run iDAR from the command line without the `-I` flag.

`Syslog is not available on Windows NT platform`

Administrator tried to configure syslog (in `ids-proxy-sch-LogProperty`) on Windows NT.

Numbered Errors

These errors are generated by iDAR; each is distinct.

```
100203 Missing second quote in RDN AVA
100301 Too many AVAs in RDN (%d)
100302 Problem decoding RDN component
100303 Extra quote in RDN
100304 Problem decoding RDN component
```

Badly formed RDN component.

`110201 Could not open %s`

Could not open the initial tailoring file.

`110202 Line %d too long/missing newline`

Each line in the initial tailoring file must be terminated with a new line, or the line is more than 1024 characters long.

`110203 Missing colon in line %d`

Each option / value pair in the initial tailoring file must be separated by a colon.

```
110204 Unrecognized keyword in line %d (%s)
110205 Problem parsing line %d (%s)
```

Bad option name in initial tailoring file.

`110207 Unknown configuration_url type %s`

The protocol selector must be either `ldap` or `file` for `configuration_url` in the initial tailoring file.

`110303 Problem parsing (%s)`

One of the attribute values in the `ids-proxy-sch-GlobalConfiguration` entry was bad.

`110401 Unknown property object %s referenced from global entry`
`110402 Unknown property object %s referenced from global entry`

The property being referenced is incorrectly specified. This could mean either the property entry is incorrectly specified or it is not being included correctly. Run iDAR in debug mode and make sure all the properties are added. Make sure the spelling of the property name is correct at the point of inclusion.

`110403 Unknown property type %s referenced from %s`

`110404 Unknown property type %s referenced from %s`

Only allowed property from `ids-proxy-sch-GlobalConfiguration` object class is `ids-proxy-sch-LogProperty`.

`130108 ids-proxy-con-ava has attribute name specified twice in %s in %s`

An attribute name was specified twice.

`150201 SetServiceStatus call failed`

Windows NT service manager configuration problem.

`150304 Failed to Reread Configuration... Aborted`

iDAR tried to reread its config files and failed.

`150403 iDAR process %d has exited, attempting restart in 10 seconds`

The monitor has detected that iDAR has exited abnormally and will attempt to restart it.

`170802 Could not decode sasl cred`

iDAR could not decode the credentials received from the client as part of a SASL bind.

`170910 Operation received while SASL bind in progress`

No operations are allowed when a SASL bind is in progress. The client sent a request other than “bind” while SASL bind was in progress.

`170911 Duplicate message id received: %d`

The client sent two simultaneous requests with the same message id.

`170912 Operation received between SSL negotiation`

No operations are allowed when the START TLS is in progress. The client initiated START TLS but send other operations before completing it.

`171012 Could not establish SSL for referral server`

SSL could not be established with a LDAP server to which iDAR was referred.

`171027 OnBindSuccess rule failed... closing connection`

An error occurred while evaluating `OnBindSuccess` rule. Note: This error message does not mean `OnBindSuccess` rule evaluated to `FALSE`.

`190201 Wrong number of elements in NOT filter`

Bad filter.

```
190402 First tag was not a SEQUENCE: source(%s, %d)
190403 First element was not a MessageID: source(%s, %d)
```

A incorrectly formatted PDU was received from the client or the server.

```
210302 ber_flush failed: Destination(%s, %d)
210303 ber_flush failed: Destination(%s, %d)
```

iDAR failed to send data to the server or the client because either the connection was reset or other network problems exist.

`240101 Failed to tailor entry %s for attribute %s a bad attribute was found in a Group entry.`

`240102 Either (or both) LoadBalanceProperty or ids-proxy-con-server are present in %s`

One of the above is allowed.

`240201 A Property failed check from group %s`

One of the properties included in this group was inconsistent with the Group.

`240701 Could not find property %s referenced from %s`

There can be several reasons for this. You must reference a property by its name, i.e., the value of the attribute `ids-proxy-con-Name` in the property entry you are referencing. Check to see if the spelling is correct. The next thing to check is if the referred property is one level under the value of `ids-proxy-sch-PropertyBase` attribute in the `ids-proxy-sch-LDAPProxy` object entry. Then make sure if the referred property belongs to `ids-proxy-sch-belongs-to` the correct server instance and is enabled (`ids-proxy-sch-enable`).

`240703 Wrong property type for ids-proxy-con-server in %s`

The property type can only be `ids-proxy-sch-LDAPServer`.

`280205 Could not find property %s referenced from %s`

See `240701`.

```
280302 Check failed of one of the server properties
```

One of the properties included in this property was inconsistent with the Property.

```
300301 Connection from %s has IP options present
300302 Could not clear IP options from %s
300402 Failed for %s,can't verify hostname
```

iDAR could not resolve a hostname.

```
300403 Host name/name mismatch %s
300404 Host name/address mismatch %s
```

A reverse lookup of the host yielded a different host name for the client.

```
301101 Failed
```

iDAR failed to send data to a client / server.

```
330207 Failed
330211 Failed
```

Call to listen for failed.

```
340101 Failed to open connection, #times tried %d
```

Could not open connection to configuration LDAP server.

```
340901 No file name specified
```

No configuration file name was found in startup configuration file.

```
340902 No Server Entry DN Specified
```

The DN for the server's entry (`object class ids-proxy-sch-LDAPProxy`) was missing from the startup configuration file.

```
340912 No proper type for Server Entry %s
```

The server entry must have an objectclass attribute with value `ids-proxy-sch-LDAPProxy`.

```
380101 Failed
380103 Failed
```

A thread could not be created.

```
380201 Failed
380202 Failed
```

A mutex lock could not be acquired.

`510103 No group %s as referenced in %s found`

See explanation of 240701. Similar logic applies to Groups.

`520112 Contradiction between ids-proxy-con-use-version`

The LDAP protocol version that iDAR is configured to use with a backend LDAP server must be supported by that backend server.

Appendix C

iDAR FAQ, Features, and Troubleshooting

This appendix contains useful information on iPlanet Directory Access Router (iDAR). It contains answers for frequently asked questions (FAQs), clarifications on certain iDAR features, and troubleshooting information.

The appendix has the following sections:

iDAR FAQ

What is iDAR?

iDAR is an LDAP proxy for LDAP clients and LDAP servers. Requests from LDAP clients are forwarded to LDAP servers based on rules defined in iDAR's configuration. Results from the server are passed back to clients, again based on rules defined in the configuration. This process is totally transparent to the clients, which connect to iDAR just as they would to any LDAP server.

Why do I need an LDAP Proxy Server?

Many enterprises want to make some part of their directory information externally visible, while keeping other parts internally private. With iDAR you can accomplish this goal easily, and without assigning directory passwords to external clients. iDAR can also be used as a high availability solution for enterprise directory service with load balancing and failover features.

Additional security features such as protection from denial of service attacks and search limits are also provided.

What version of the LDAP protocol does iDAR support?

iDAR supports LDAP clients or chaining LDAP servers that use either the LDAPv2 or the LDAPv3 protocol.

Does iDAR support secure authentication and encryption?

iDAR supports SSLv3 services for public-key based data encryption using certificates. Secure authentication and encryption available to LDAP clients can either use the secure LDAP port or the Internet Transport Layer Security (TLS) model, which uses the Diffie-Hellman, Digital Signature Standard (DSA), and Triple-DES algorithms.

Does iDAR work with any LDAP-enabled directory server?

iDAR will work with any LDAP-conformant directory server. Some directory product vendors claim to implement LDAP in their marketing literature, but the reality is often a different story. iDAR has been the most thoroughly tested with the iPlanet Directory Server.

If the iDAR 5.0 Console is used, the iPlanet Directory Server 5.0 is the supported configuration depository.

Is there a configuration utility available to configure iDAR?

iDAR 5.0 includes a Java-based GUI (console) that can be used to configure iDAR. The console uses the iPlanet Directory Server to store the configuration it generates.

iDAR Features

Can iDAR be used to prevent denial-of-service attacks?

Yes. You can limit the number of simultaneous operations processed per connection, number of operations allowed per connection, total number of concurrent connections, maximum concurrent connections per defined group (network, subnetwork or based on bind DN), and maximum concurrent connections for a single IP address.

Does iDAR support “reverse” proxying?

In a strict sense, iDAR is a reverse proxy; however, the LDAP protocol does not support the concept of reverse proxying.

Can iDAR be used to prevent “trawling” of an LDAP directory?

Yes. Trawling refers to very broad queries designed to download large portions of your directory, a practice many sites wish to prohibit. iDAR can prohibit or limit trawling in a number of ways:

- The scope of searches can be limited to a single level of the directory tree, entire subtrees can be hidden, and a hard limit on the number of entries returned in response to a query can be set.
- Inequality searches can be forbidden, thus disallowing searches that return many results based on exclusion and substring searches can be restricted by length; for example, prohibiting searches for all entries with a surname starting with the letters A-Z.
- iDAR can also be configured to deny un-indexed searches. Un-indexed searches are inefficient and can possibly have a negative impact in performance.

Does iDAR do automatic load balancing of queries?

iDAR supports automatic server load balancing among a set of back-end LDAP servers. iDAR also supports automatic fall-over to a secondary LDAP server if the primary LDAP server is down.

How many LDAP servers can one iDAR server load balance?

The performance needs of the directory server and the complexity of work being done by iDAR determines the optimal number of directory servers that iDAR should load balance. For example, if iDAR is doing complicated work, such as attribute renaming, the number of directory servers iDAR is configured to load balance should be reduced. Consider adding more iDAR units to compensate for possible performance impacts of complex iDAR configurations.

Can search requests be filtered?

Yes. You can configure iDAR to refuse searches that attempt to search on a particular attribute. In addition, you can configure iDAR to modify incoming search requests to conform to a designated minimum search base, search scope, and time limit.

Can search results be filtered?

Yes. Results can be filtered both in terms of number of entries returned and the attributes that are included in the result set. Search result entries can also be filtered based on the entry DN or content.

How are access groups defined?

Varying levels of access to the directory are provided to clients based on the network address of the client. Thus, different levels of access can be granted to clients outside the corporate firewall, inside the firewall, on the executive subnetwork, and even to individual machines. Further, access level can be changed upon a successful completion of a LDAP Bind operation by the client or when a SSL session is established.

Does iDAR support protected password authentication?

Yes. Through the use of the SASL mechanism a variety of protected password authentication schemes can be implemented. These mechanisms must be supported by the back-end directory server. iDAR does not support SASL mechanisms with connection protection and SASL EXTERNAL mechanism.

Does iDAR automatically follow referrals?

The following of referrals is configurable based on access group. Various access groups can be configured to automatically follow referrals, return referrals, or discard referrals.

Does iDAR cache search result information?

iDAR 5.0 does not support search result caching.

Can iDAR do attribute renaming?

iDAR can transparently rename attribute names between clients and servers.

Troubleshooting

How can I analyze logs of connection attempts?

iDAR can be configured to either use `syslog` or write to a specified log file. A popular UNIX utility known as swatch is freely available from Stanford University (`ftp://ftp.stanford.edu/general/security-tools/swatch`). Swatch can be used to monitor the log files generated by iDAR and to notify the administrator when defined events occur.

I have configured iDAR to follow referrals. However, when I perform a search with a LDAPv2 client I get error 32 (No such object) or some other error.

In order for iDAR to receive referrals from the back-end servers, it must use LDAPv3. Make sure you have selected “LDAP version 3 only” on LDAP version to use selection for each of your LDAP server properties.

iDAR is load balancing across ‘n’ servers. When one of these servers go down, some of my clients seem to hang.

Make sure that the server in question is reachable from the host iDAR is running on. An unreachable server (caused by bad network or a hung server) can cause iDAR to wait on a long timeout. This may cause the clients that are virtually connected to the unreachable server to appear to be hung. However, a server that is not running should be detected quickly via “Connection refused.” If clients still appear to be hanging under these conditions, then you should verify that ICMP messages can be exchanged between the host on which the server is supposed to be running and the host on which the iDAR is running. A firewall may block these messages.

I notice in the log files that some idle client connections are routinely failed over even though all my back-end servers are up.

Your back-end directory server is timing out idle connections and closing them. iDAR fails over these closed connections. You must set an idle connection timeout for iDAR as well. This will clean up idle and leaked client connections and also guard against one form denial of service attack.

Is there a way to restrict search requests containing the presence filter?

iDAR 5.0 does not have any direct mechanisms to restrict clients from using the presence filter. There are two indirect ways to address this issue.

You can set the `ids-proxy-con-forbidden-compare` to the name of the attribute that you do not want to be compared. This method is over restrictive, as it will reject searches containing both (mail=*) and (mail=Andy*) filters.

On the other hand, since presence filters (attrName=*) always generate the same result (assuming the data did not change), we can use the ids-proxy-con-size-limit attribute and the `ids-proxy-sch-SizeLimitProperty` to limit the damage. Although, LDAP does not require entries to be returned in a given order, under most (all) implementations, the set of result will either be returned in sorted order or unsorted order, and these will be the same every time. Hence, if iDAR is configured with a size limit, (using the size-limit attribute or the SizeLimitProperty) only the first 'n' of these sets will be returned every time. Because there can only be two sets of these 'n' entries, the risk of trawling the directory is greatly reduced.

Note that iDAR tries to set this size limit in the request itself when possible, and therefore the directory server will not be burdened with sending all the entries.

The size limit property gives you the option of applying exceptions to size limits imposed when necessary. Suppose, for example, that you have an entry of o=A, under which there are 400 organization units. Under each of those OU's there are people. If you want clients to see all the OU's but only see 5 people at a time, you can set up the `SizeLimitProperty` such that no limit is applied for a search with base o=A and one level scope. For all other searches a limit of 5 applies.

I am using JNDI. iDAR rejects some (not all) of my search queries saying it could not decode search request. If I send the query directly to my directory server, it works fine.

iDAR is more strict about attribute type as defined in LDAP standards. If you are requesting specific attributes to be returned as part of that search result, make sure that the attribute type names conform the LDAP standards stated in RFC 2251 section 4.1.4, paragraph 3.

"A specification may also assign one or more textual names for an attribute type. These names MUST begin with a letter, and only contain ASCII letters, digit characters and hyphens. They are case insensitive. (These ASCII characters are identical to ISO 10646 characters whose UTF-8 encoding is a single byte between 0x00 and 0x7F.)"

JNDI is known to send "" strings attribute names as part of attribute list to be returned for a search. As stated above empty strings are illegal LDAP attribute type names.

When I try to execute a task or perform some console function, I get an error message saying I need to make sure the Administration Server is running properly and that this host is permitted to connect to the Administration Server.

Log in to the Administration Server that is managing the iDAR whose console produced the error messages. It may be necessary to start the iPlanet Console on the host machine of the Administration Server. Open the server console for the Administration Server that is managing the iDAR on which you are unsuccessfully trying to invoke tasks. Click the Configuration tab and then the Network tab. Under Connection Restrictions, make sure that the host machine of the iPlanet Console that is unsuccessfully trying to manage iDAR is not restricted from accessing the Administration Server. See the iPlanet Console *Server Management Guide* for more information.

Index

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V