

How To Configure OpenLDAP to work with Check Point Devices

OPSEC Engineering
May 2006



Check Point protects every part of your network—perimeter, internal, Web—to keep your information resources safe, accessible, and easy to manage.

Abstract	4
Overview	4
Environment.....	4
Note on Environments	5
Procedural Outline	5
Procedure	6
Install Red Hat Enterprise Linux	6
Configure the OpenLDAP Server.....	6
Define the root of the directory server.....	6
Define the directory manager.....	7
Add the Check Point Schema definition.....	7
Starting the OpenLDAP server	8
Setting up the initial directory content.....	8
Testing the directory with ldapsearch.....	9
Testing the directory with LDAP Browser	9
Install and Configure VPN-1	10
Outline.....	10
Install and Configure Connectra	15
Configuring an LDAP Account Unit	16
Creating an LDAP User Group For Authorization	16
Further configuration options	17
Encrypted communication	17
Create SSL certificate	18
Configure OpenLDAP to use SSL.....	19
Configure VPN-1to use LDAP over SSL.....	19
Restricting access.....	19
Creating accounts for the firewall and the administrator.....	19
Setting access rights in slapd.conf	20
Configure VPN-1to use multiple LDAP accounts.....	21
Tips, Tricks and Troubleshooting.....	22
OpenLDAP 2.1.x and later.....	22
Migrating an existing OpenLDAP directory to be used by FireWall-1	22
Trying to fetch branches fails	22
How to check whether a LDAP license (Account Management Client) is installed	23
How to increase search speed when using dynamic groups	23
Size limit exceeded	24
Resources	25
OpenLDAP and LDAPv2	25
OpenLDAP and LDAPv3	25
ObjectClass Violations.....	25
OpenLDAP and IPv6	25
OpenLDAP and LDAP over SSL	25
VPN-1 schema extension.....	26

Appendix..... 26
Netscape LDIF Schema to OpenLDAP schema conversion..... 26
Check Point schema extension file for OpenLDAP (fw1ng.schema)..... 26

Abstract

Check Point™ VPN-1® NG and Connectra™ have the ability to access LDAP directory servers for managing users, groups and templates. OpenLDAP is a free, stable, and widely used LDAP Server on UNIX platforms. This guide describes how to configure OpenLDAP on Red Hat Linux for the integration with VPN-1 NG SmartDirectory and Connectra.

Document Title: How to configure OpenLDAP with Check Point Devices

Creation Date: 14-Jan-2003

Modified Date: 01-May-2006

Document Revision: 1.9

Product Class: FireWall-1 / VPN-1, Account Management Client

Product and Version: FireWall-1/VPN-1 NG, Connectra NGX, OpenLDAP 2.x

Typeface or Symbol	Meaning	Example
AaBbCc123	The names of commands, files, and directories; onscreen computer output.	Open your slapd.conf file (located at /etc/openldap/) and locate the database section.
AabBcC123	What you type, when contrasted with on-screen computer output.	[root@redhat root]# rpm -qa grep ldap openldap-2.0.25-1 openldap-servers-2.0.25-1 openldap-clients-2.0.25-1 [root@redhat root]#
<i>AaBbCc123</i>	Book titles or words to be emphasized.	Please refer to the <i>VPN-1 Getting Started Guide</i> for further information.
AaBbCc123	Text that appears on an object in a window	Create a new LDAP Account Unit using the objects tree or the menu (using Manage/Servers).

Overview

This paper outlines how to configure the OpenLDAP server on a Red Enterprise Linux system for use with VPN-1 NG and Connectra. This includes the configuration of the OpenLDAP server itself as well as the integration of the VPN-1schema extensions. There is also a brief overview of how to configure VPN-1 and Connectra for using directory services.

Environment

Throughout this paper the following components are involved:
 Check Point NGX Management Server and Enforcement Module
 Check Point NGX SmartConsole Clients
 Connectra NGX

Red Hat Enterprise Linux
Freeware tools used:

OpenLDAP 2.3.XX or later - <http://www.openldap.org>
Berkeley DB 4.2.52 or later - <http://www.sleepycat.com>
OpenSSL 0.9.7e or later – <http://www.openssl.org>

Note on Environments

The OpenLDAP version included in Red Hat Enterprise Linux is not the latest one available. The focus of this paper is to use the package included in the Red Hat distribution. There are more recent binaries available and there is also the source code package. The latest stable version 2.3.20 and the default Red Hat package 2.0.27 were tested.

The LDAP directory structure used in this paper is very simple and concentrates only on the usage for the Check Point devices. Directories used in production environments are normally more complex in their overall structure and contain much more information than the one used in this paper. This may also include the use of multiple directory servers providing the same information (using directory replication) for fault tolerance. This kind of configuration is similar to a configuration with just one directory server. This scenario is just mentioned in this paper and not described in detail.

The firewall requires an Account Management license to manage users on LDAP servers. This license may be purchased separately or be included in a bundle (e.g. SmartCenter Pro). Please make sure that you have such a license before starting (see [How to check whether a LDAP license \(Account Management Client\) is installed](#)).

The backend used in the following configs is ldbm. Your database may differ. See the [OpenLDAP FAQ on backends](#).

Procedural Outline

The following steps are necessary to use an OpenLDAP Server with FireWall-1:

1. Install Red Hat Enterprise Linux
 - a. The OpenLDAP packages are not installed by default. Please make sure to select the OpenLDAP client and server packages for installation. Red Hat EL3 includes OpenLDAP 2.0.27-XX rpm and Red Hat EL4 includes OpenLDAP 2.2.13-XX rpm packages.
2. Configure the OpenLDAP Server
 - a. Define the root of the directory server
 - b. Define the directory manager
 - c. Add the VPN-1Schema definition
3. Starting the OpenLDAP server
4. Setting up the initial directory content
5. Testing the directory

Note: This paper does not focus on the task of how to install Red Hat Linux for the OpenLDAP server or how to install and initially configure SecurePlatform with FireWall-1 or Connectra. Please consult the appropriate documentation for these tasks. Please refer to the *User Management* Guide for further information how to integrate user databases on LDAP Servers.

Procedure

Install Red Hat Enterprise Linux

Please install Red Hat Linux with a configuration that matches your environment and your specific needs. Please make sure to install the OpenLDAP packages. You can choose this during the installation process or manually install it using rpm afterwards.

To make sure everything is installed an rpm -qa should at least output the following packages:

```
openldap
openldap-servers
openldap-clients
```

```
[root@redhat root]# rpm -qa | grep ldap
openldap-2.0.25-1
openldap-servers-2.0.25-1
openldap-clients-2.0.25-1
```

Configure the OpenLDAP Server

In this step we will change the default OpenLDAP configuration file (located at /etc/openldap/slapd.conf) to reflect our configuration and organization. Please make sure to make a backup of this file and to store it in a safe location. The organization used in this paper is called "Example Labs". Example Labs has an Internet domain called example.com. We will use this later to define the structure of our directory.

Note: In this paper we will use domain components to create the directory tree (e.g. dc=example, dc=com). You may see other directories that are using an organization unit/organization/country scheme (e.g. o=example,c=de). It makes no difference for the firewall how the directory tree is constructed. Nevertheless the form with the domain components seems to gain more popularity because it allows clients to discover directory servers using DNS.

Define the root of the directory server

Open the OpenLDAP configuration file (located at /etc/openldap/slapd.conf) and locate the database section. It looks similar to this. We will change/delete parts of the example file step by step to match our organization (marked bold). Lines starting with a "#" are comment lines and can be ignored.

```
database ldbm
suffix "dc=my-domain,dc=com"
#suffix "o=My Organization Name,c=US"
rootdn "cn=Manager,dc=my-domain,dc=com"
#rootdn "cn=Manager,o=My Organization Name,c=US"
# Cleartext passwords, especially for the rootdn, should
# be avoided. See slappasswd(8) and slapd.conf(5) for details.
# Use of strong authentication encouraged.
# rootpw secret
# rootpw {crypt}ijFYnCSNctBYg
```

The root of our directory will be dc=example,dc=com. Please change the suffix line accordingly:

```
database ldbm
suffix "dc=example,dc=com"
```

```

rootdn "cn=Manager,dc=my-domain,dc=com"
#rootdn "cn=Manager,o=My Organization Name,c=US"
# Cleartext passwords, especially for the rootdn, should
# be avoided. See slapd.conf(5) for details.
# Use of strong authentication encouraged.
# rootpw secret
# rootpw {crypt}ijFYncSNctBYg

```

Define the directory manager

The directory manager has always full read/write access to the directory. Although it is possible to define the directory manager's password in clear text this is not recommended. We will use `slappasswd` to create encrypted passwords using CRYPT as the scheme. Consult the `slappasswd` man page if you would like to use a scheme other than CRYPT.

```

[root@redhat root]# slappasswd -h {CRYPT}
New password: <Your Password>
Re-enter new password: <Your Password>
{CRYPT}hHNhi1RUKLV6M
[root@redhat root]#

```

Change the `rootdn` and `rootpw` in `slapd.conf` to reflect your directory manager's name and the just created password:

```

database ldbm
suffix "dc=example,dc=com"
rootdn "cn=Manager,dc=example,dc=com"
rootpw {CRYPT}hHNhi1RUKLV6M

```

Add the Check Point Schema definition

By default OpenLDAP does not include the Check Point schema. Without any further modification it is not possible to create users with any kind of Check Point specific attributes (You will receive an error message like "objectclass violation" if you try this).

Note: The `fw1person` objectClass is defined as an AUXILIARY rather than a STRUCTURAL type so that existing users can be managed via SmartDirectory. It wouldn't be possible to edit existing users which have a `structuralObjectClass` value of `inetOrgPerson` if `fw1person` was STRUCTURAL because `inetOrgPerson` is superior to `fw1person`. SmartDirectory adds the objectClass `fw1person` when modifying users. This would break the "chain of superiors" if `fw1person` was a STRUCTURAL type.

Check Point provides schema extensions for Netscape Directory servers and for Microsoft Active Directory (see *User Management Guide* for more information). Neither of the two can be used directly for OpenLDAP. But OpenLDAP uses a simple format for schema definitions, which allows us to create the OpenLDAP schema from the Netscape Schema (located at `$FWDIR/lib/ldap/schema.ldif`). This can be done using a [simple shell script](#), which can be found, at the end of this paper. Alternatively you can use the already created [OpenLDAP schema](#), which can also be found at the end of this paper. Copy the Check Point schema definition file to `/etc/openldap/schema/fw1ng.schema` and add an appropriate line to `slapd.conf`.

```

include /etc/openldap/schema/core.schema
include /etc/openldap/schema/cosine.schema
include /etc/openldap/schema/inetorgperson.schema
include /etc/openldap/schema/fw1ng.schema

```

Starting the OpenLDAP server

The OpenLDAP server is configured now and we are ready to start it. Before starting the OpenLDAP server automatically at boot time we should make sure that everything works fine. Start the OpenLDAP server manually (in debug mode) and check whether it runs smoothly. Use **Control-C**, “**^C**” to terminate it.

```
[root@redhat root]# /usr/sbin/slapd -d 768
daemon: socket() failed errno=97 (Address family not supported by protocol)
slapd starting
^C
slapd shutdown: waiting for 0 threads to terminate
slapd stopped.
```

As you can see slapd complains about a non-supported address family. Please refer to the [IPv6 section](#) under [Troubleshooting](#) for an explanation how to check whether your system supports IPv6. In most cases this can be safely ignored.

Now you can check whether OpenLDAP can be started/stopped without errors using the service script.

```
[root@redhat root]# service ldap start
Starting slapd: [ OK ]
[root@redhat root]# service ldap stop
Stopping slapd:
```

It's also possible to check the state of the OpenLDAP server using service ldap status.

```
[root@redhat root]# service ldap status
slapd (pid 849 848 844) is running...
```

If everything works fine you can permanently start the OpenLDAP server at boot time. Use chkconfig to configure **ldap** to start at boot time.

```
[root@redhat root]# chkconfig --list ldap
ldap          0:off  1:off  2:off  3:off  4:off  5:off  6:off
[root@redhat root]# chkconfig ldap on
[root@redhat root]# chkconfig --list ldap
ldap          0:off  1:off  2:on   3:on   4:on   5:on   6:off
```

Setting up the initial directory content

The OpenLDAP server is completely configured now and waiting for data. But although we defined the root suffix in slapd.conf the directory database is still empty. We still need to fill it with initial data. This initial data tells the directory server which kind of object the root and the directory manager are. Prepare an initial LDIF file, initial.ldif, like the one below and import it using ldapadd.

```
# Organization for Example Labs
dn: dc=example,dc=com
objectClass: dcObject
objectClass: organization
dc: example
o: Example Labs
description: Example Labs
#
# Organizational Role for Directory Manager
dn: cn=Manager,dc=example,dc=com
```

```
objectClass: organizationalRole
cn: Manager
description: Directory Manager
```

```
[root@redhat root]# ldapadd -D "cn=Manager,dc=example,dc=com" -f initial.ldif -h
127.0.0.1 -W -x
Enter LDAP Password: <Your Password>
adding new entry "dc=example,dc=com"
adding new entry "cn=Manager,dc=example,dc=com"
```

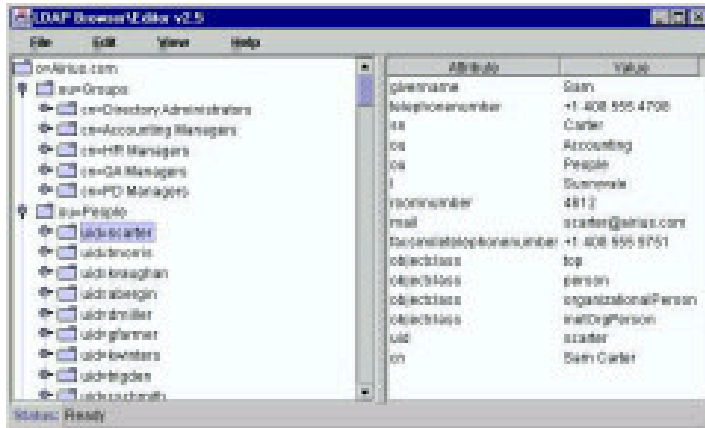
Testing the directory with ldapsearch

We just added data to the directory. Now we want to try to query the OpenLDAP server. The easiest way is using ldapsearch:

```
[root@redhat root]# ldapsearch -h 127.0.0.1 -D "cn=Manager,dc=example,dc=com" -W -
x -b "dc=example,dc=com" '(objectclass=*)' '*'
Enter LDAP Password: <Your Password>
version: 2
#
# filter: (objectclass=*)
# requesting: *
#
# example, lab
dn: dc=example,dc=com
objectClass: dcObject
objectClass: organization
dc: example
o: Example Labs
description: Example Labs
# Manager, example, lab
dn: cn=Manager,dc=example,dc=com
objectClass: organizationalRole
cn: Manager
description: Directory Manager
# search result
search: 2
result: 0 Success
# numResponses: 3
# numEntries: 2
```

Testing the directory with LDAP Browser

Another possibility is to use a graphical tool like Jarek Gawor's LDAP Browser (<http://www.iit.edu/~gawojar/ldap/>). It allows you to graphically explore your directory server's content. It's also possible to use it to import the initial.ldif file mentioned above.



Querying OpenLDAP with LDAP Browser

If you plan to use the LDAP Browser to import the initial.ldif file and to test the directory you might not need the `openldap-client` package installed because `ldapadd` and `ldapsearch` aren't necessary anymore.

Install and Configure VPN-1

Please refer to the VPN-1 *Getting Started* Guide for further information.

Outline

1. Create an OpenLDAP_DS profile (optional).
2. Enable SmartDirectory in Global Properties.
3. Create a host object for the OpenLDAP server.
4. Create an LDAP Account Unit.
5. Begin Managing Users, Groups, and Templates in SmartDirectory

Create an OpenLDAP_DS Profile (optional)

The LDAP profile is selected and applied during the creation or modification of an LDAP Account Unit. You can modify certain parameters of the default profiles to match the directory structure of your LDAP server. This is done either by modifying an existing LDAP profile or by copying the settings of an existing LDAP profile into a newly created profile and then modifying the profile to fit the requirements.

The `objects.C` file contains four profiles, each corresponding to a SmartDirectory (LDAP) server type. These are `Microsoft_AD`, `Netscape_DS`, `Novell_DS`, and `OPSEC_DS`. You can use the `OPSEC_DS` profile or create a new `OpenLDAP_DS` profile using `dbedit` and the following file. Close all SmartConsole applications and then execute the following.

Note: To modify existing objects it is best to use the `GuidbEdit` tool. The `dbedit` commands below are offered as guides for modifying the LDAP profile. Manually verify the changes after invoking the file using the `GuidbEdit` tool.

```
# dbedit -local -u <administrator> -p <password> -f OpenLDAP_DS.dbedit -r
"add OpenLDAP_DS profile"
#
#
```

```

# Add OpenLDAP_DS ldap profile
# Use the command
# dbedit -local -u <administrator> -p <password> -f OpenLDAP_DS.dbedit \
# -r "add OpenLDAP_DSprofile"
#
create LDAP_policy OpenLDAP_DS
modify ldap OpenLDAP_DS Common:ExpirationDateAttr fw1expiration-date
modify ldap OpenLDAP_DS Common:MainVersion 4
modify ldap OpenLDAP_DS Common:Vendor OpenLDAP
addelement ldap OpenLDAP_DS Read:BranchObjectClass organization
addelement ldap OpenLDAP_DS Read:BranchObjectClass domain
addelement ldap OpenLDAP_DS Read:DomainObjectClass domain
addelement ldap OpenLDAP_DS Read:GroupMembershipAttr uniqueMember
addelement ldap OpenLDAP_DS Read:GroupObjectClass groupOfUniqueNames
addelement ldap OpenLDAP_DS Read:UserObjectClass organizationalPerson
addelement ldap OpenLDAP_DS Read:UserObjectClass inetOrgPerson
addelement ldap OpenLDAP_DS Read:UserObjectClass fw1person
addelement ldap OpenLDAP_DS Write:DomainObjectClass domain
rmelement ldap OpenLDAP_DS Write:GroupObjectClass groupOfNames
addelement ldap OpenLDAP_DS Write:GroupObjectClass "groupOfNames:member"
addelement ldap OpenLDAP_DS Write:GroupObjectClass
"groupOfUniqueNames:uniqueMember"
addelement ldap OpenLDAP_DS Write:UserObjectClass organizationalPerson
addelement ldap OpenLDAP_DS Write:UserObjectClass inetOrgPerson
addelement ldap OpenLDAP_DS Write:UserObjectClass fw1person
update ldap OpenLDAP_DS

```

Profile Attributes

The attributes that are commonly configured by category are described here.

attribute	category C=common R=read W=write	default	other
UserLoginAttr	C	uid	One value allowed
UserPasswordAttr	C	userPassword	One value allowed
TemplateObjectClass	C	fw1template	Multiple values allowed
ExpirationDateAttr	C	fw1expiration-date	One value allowed
ExpirationDateFormat	C	CP format is yyyyymmdd	One value allowed
PsswdDateAttr	C	fw1pwdLastMod	One value allowed
PsswdDateFormat	C	CP format is yyyyymmdd	One value allowed
BadPwdCountAttr	C	fw1BadPwdCount	One value allowed
ClientSideCrypt	C	<ul style="list-style-type: none"> • 0 for most servers • 1 for Netscape_DS 	One value allowed

		if not using encrypted password, SSL is recommended	
DefaultCryptAlgorithm	C	<ul style="list-style-type: none"> • Plain (for most servers) • Crypt (for Netscape_DS) • SHA1 	One value allowed
CryptedPasswordPrefix	C	{Crypt} (for Netscape_DS)	One value allowed
PhoneNumberAttr	C	internationalisednumber	One value allowed
AttributesTranslationMap	C	none	Multiple values allowed
ListOfAttrsToAvoid	C	There are no values by default. In case the SmartDirectory (LDAP) server was not extended by the Check Point schema, the best thing to do is to list here all the new Check Point schema attributes	Multiple values allowed
BranchObjectClass	R	organization organizationalUnit domain (most servers)	Multiple values allowed
BranchOCOperator	R	One	One value allowed
OrganizationObjectClass	R	organization	Multiple values allowed
OrgUnitObjectClass	R	organizationalUnit	Multiple values allowed
DomainObjectClass	R	domain	Multiple values allowed
UserObjectClass	R	person organizationalPerson inetOrgPerson fw1person (most servers)	Multiple values allowed
UserOCOperator	R	One	One value allowed
GroupObjectClass	R	Groupofnames Groupofuniquenames (most servers)	Multiple values allowed
GroupOCOperator	R	One	One value allowed
GroupMembership	R	<ul style="list-style-type: none"> • Member mode defines the member DN in the Group object (most servers) • MemberOf mode defines the group DN in the member object (in Microsoft_AD) • Both mode defines both the member DN in the Group object and the group DN in the Member object. 	One value allowed
UserMembershipAttr	RW	MemberOf	One value allowed
TemplateMembership	RW	<ul style="list-style-type: none"> • Member mode defines the member DN in the Group object (most servers) • MemberOf mode defines the group DN in the member object (in 	One value allowed

		Microsoft_AD)	
TemplateMembershipAttr	RW	member	Multiple values allowed
UserTemplateMembershipAttr	RW	memberoftemplate	Multiple values allowed
OrganizationRDN	W	o	One value allowed
OrgUnitRDN	W	ou	One value allowed
UserRDN	W	cn	One value allowed
GroupRDN	W	cn	One value allowed
DomainRDN	W	dc	One value allowed
AutomaticAttrs	W	user:userAccountControl:66048 For Microsoft_AD This means that when a user object is created an extra attribute is included automatically: userAccountControl with the value 66048	Multiple values allowed
GroupObjectClass	W	groupOfNames:member groupOfUniqueNames:uniqueMember	Multiple values allowed
OrgUnitObjectClass	W	organizationalUnit	Multiple values allowed
OrganizationObjectClass	W	organization	Multiple values allowed
UserObjectClass	W	person organizationalPerson inetOrgPerson fw1Person	Multiple values allowed
DomainObjectClass	W	domain	Multiple values allowed

Enable SmartDirectory in Global Properties.

By default User Management on LDAP Servers is disabled. To use User Management on LDAP Servers activate LDAP Account Management in the **Global Properties**. The Global Properties can be reached via a toolbar button or using **Policy/Global Properties**.

Create a new Host Object

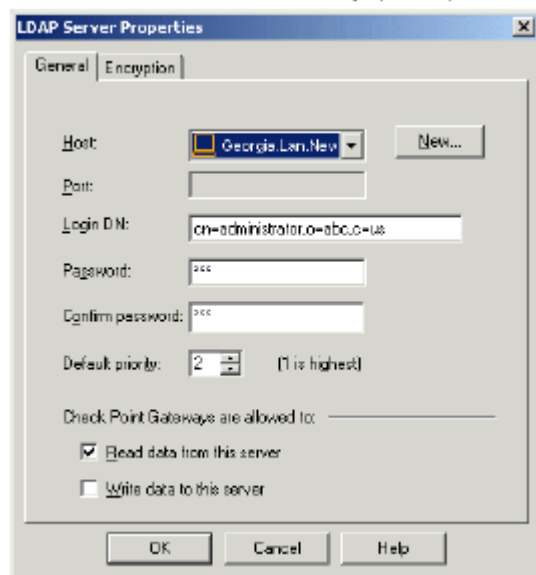
Create a new host object for the host where the OpenLDAP server is running using **Manage/Network Objects/New... Node - Host**.

Create a new LDAP Account Unit

Create a new LDAP Account Unit using **Manage/Servers/New... LDAP Account Unit**. The LDAP Account Unit Properties window consists of several tabs. Enter a name and choose a profile for the directory.

1. In the **General** tab define the general settings of the SmartDirectory (LDAP) Account Unit. Decide whether this Account Unit is to be used for CRL retrieval, user management or both. For the purposes of this paper select User management. Select a profile to be applied to the new Account Unit. For OpenLDAP servers use the profile **OPSEC_DS** or **OpenLDAP_DS**.
2. In the **Servers** tab, display the SmartDirectory (LDAP) servers to be used by the Account Unit. The order in which they are displayed is also the default order in which they will be queried. This priority can be defined per Gateway and in addition it can be defined on the Account Unit. Add a server to the displayed list. For backwards compatibility, select a SmartDirectory (LDAP) server which is able to work with pre-NG FP3 VPN-1/FireWall-1 modules.

Double-click on a SmartDirectory (LDAP) server in the displayed list in order to modify it. The **SmartDirectory (LDAP) server Properties** window is displayed.



SmartDirectory (LDAP) Server Properties window

In the **General** tab, you can modify the login DN of the SmartDirectory (LDAP) server, e.g. "cn=Manager,dc=example,dc=com". Define the priority of the selected SmartDirectory (LDAP) server. Specify a password to be used for authentication purposes. You will need to confirm this password. Select Read from ... and Write data to this server and click OK.

3. In the **Objects Management** tab, select the SmartDirectory (LDAP) server on which the objects are managed. The branches for the selected SmartDirectory (LDAP) server can be retrieved by selecting **Fetch branches**, or they can be added manually. These branches will be searched when this SmartDirectory (LDAP) server is queried.
4. In the **Authentication** tab, the authentication limitations and default authentication settings for a user on an Account Unit are defined. The Allowed Authentication schemes limits the user's authentication access only to those authentication schemes. You can allow several authentication schemes which can be applied per user, or you can apply a default authentication scheme which is applied to all the users. Users that are retrieved through this Account Unit, but which are missing authentication-related definitions, will be granted these definition using the default authentication scheme, or a complete user template. These default settings are useful, if the Check Point schema is not in place. In the

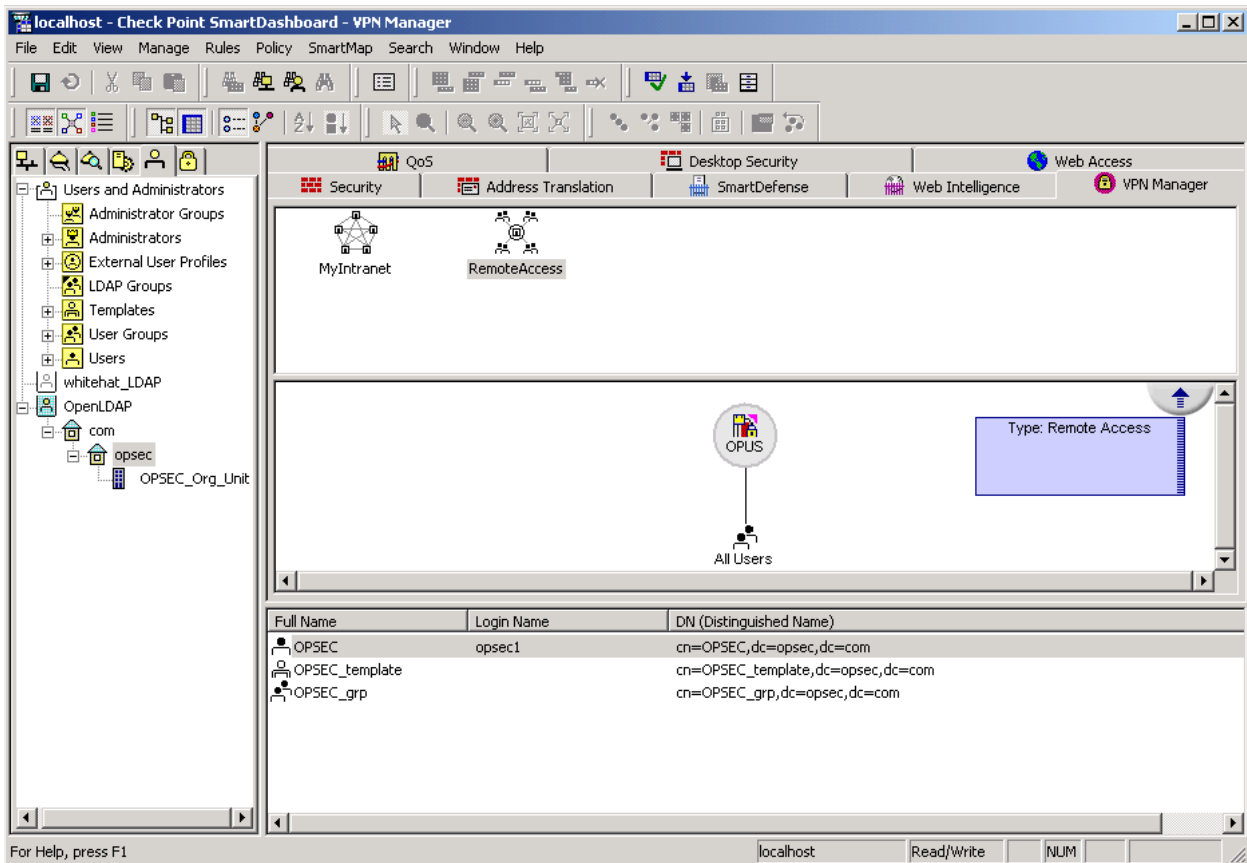
place of the Check Point schema, a user template is used to supply the authentication attributes and the other options described on this page will by default be automatically configured per user.

For all users in this Account Unit that are configured for IKE, the pre-shared secret for encryption purposes should be entered.

The number of login attempts to a user can be set, as well as the number of seconds it takes for the user's account to be unlocked, once it has been frozen.

Begin Managing Users, Groups, and Templates in SmartDirectory

Define users, groups, templates, and Organizational Units as needed. A new network object for LDAP users is created on the Users tree. (The LDAP users also appear in the objects list window to the right.)



Install and Configure Connectra

Connectra authenticates the users either through its own internal database, LDAP, RADIUS or RSA ACE/Servers and authorizes access based on Connectra group permissions.

Authentication and Authorization in Connectra

During the authentication process, the remote users are associated with one or more groups. Remote users, once authenticated, can only access those applications which have been authorized for their groups.

Authorization is the process that controls the access to the applications on the internal network. This is done by enforcing an access control policy assigned in a Connectra group and not an individual user.

Connectra requests the LDAP Server to return a list of the remote user's groups. Once the LDAP groups are mapped to the user's groups as defined on Connectra, an access control policy is assigned. Then access is given to corporate applications such as internal LAN web servers, email servers, and file shares as defined in the Connectra group.

Note: While Connectra is capable of authenticating and authorizing LDAP users and groups, it cannot administer users and groups directly on the LDAP server.

Configuring an LDAP Account Unit

1. On the navigation tree, click **Users and Groups > Authentication > LDAP account units**. The **LDAP Account Units** page appears.
2. To add an LDAP account unit, click **New**. The **Adding a new LDAP Account Unit** page appears.
3. In the **General** section, Provide the server name or IP Address of the LDAP Server and select the **LDAP type, OPSEC**. Click **OK**.
4. The **LDAP Servers** section lists your organization's LDAP servers. Each of these servers should contain the same information, allowing a redundant configuration, and thereby guaranteeing optimal system performance. To add an LDAP server, click **New**. Enter the server name and a priority. Users first attempt to connect with the server with the highest priority (1 being the highest priority). Click **OK**.
5. In the **Networking** section:
 - a. Decide whether to **Use SSL** for the connection.
 - b. Configure a port number.
6. In the **Login to LDAP** section:
 - a. Provide the Login Distinguished Name, e.g. cn=Manager,dc=example,dc=com.
 - b. Provide a password.
7. In the **Branches** section, select an existing branch, or click **New** to create a new branch.
8. Click **Apply & Fetch Branches** to display all the branches of the selected LDAP server.
9. In the **Authentication** section, select an authentication scheme, LDAP, RADIUS, or SecurID.
10. Click **Apply**.

Creating an LDAP User Group For Authorization

Connectra supports LDAP groups by mapping them to a logical group known as an "external group". Connectra interacts with LDAP servers by maintaining an "LDAP account unit" object. The administrator defines a certain point in the branch as group. Generally, the administrator needs:

- To create the users and groups on the LDAP server
- To create a local Connectra group that will then be mapped to the LDAP user group once authentication is complete.

To create an LDAP User Group:

1. On the navigation tree, click **Users and Groups > User Groups**. The **User Groups** page appears.
2. Click **New**.
3. Select LDAP User Group
4. Enter a group name and select the LDAP Account Unit created above. Select All Branches, a specific Branch, or a specific LDAP Group.
5. Click Apply.

Check Point
SOFTWARE TECHNOLOGIES LTD.
We Secure the Internet.

Check Point Connectra

Adding a new group

Users and Groups → User Groups → Adding a new group

Group | Home Page | Web Access | File Access | Mail Access | Network Access | Citrix Access

Group Details

Group name:

Comment:

LDAP Account Units:

All branches

Branch DN: ,

LDAP group DN: ,

Further configuration options

Encrypted communication

By default the communication between the Check Point device and the OpenLDAP server uses a clear text connection. To enable the use of Idaps (LDAP over SSL) you have to do some configuration work on the OpenLDAP server side as well as on the Firewall side.

Note: In this example we are going to use OpenSSL to create the key pair and the certificate needed for SSL. Please note that this paper only concentrates on the fact how to activate SSL for the Firewall. It does not deal with “real” PKI structures or functionalities. Please consult the appropriate documentation for this.

First of all we have to make sure to install SASL and OpenSSL packages. These are needed for full LDAPv3 support (take a look at the [OpenLDAP and LDAP over SSL](#) section for more information about this topic). You can choose the packages during the installation process or manually install them using rpm afterwards. To make sure everything is installed for LDAP over SSL an rpm `-qa` should at least output the following packages:

```
cyrus-sasl
cyrus-sasl-md5
cyrus-sasl-plain
openssl
```

```
[root@redhat root]# rpm -qa | grep -i -e ssl -e sasl
cyrus-sasl-2.1.7-2
openssl-0.9.6b-29
cyrus-sasl-md5-2.1.7-2
cyrus-sasl-plain-2.1.7-2
```

Create SSL certificate

To create a key pair/certificate for SSL we can use already installed make files at /usr/share/ssl/certs. Please do not use the slapd.pem file, which is already there. This file is intended for testing purposes only! Please make sure to enter your hostname as Common Name (cn). Do not use IP addresses: Most of the TLS/SSL related problems with OpenLDAP are caused by invalid Common Names. Only use the hostname. Create a slapd.pem file in /usr/share/ssl/certs using make slapd.pem. Generate and write down the fingerprint. We will need it later to verify it on the firewall side. Make sure that the created file slapd.pem is readable by the OpenLDAP daemon.

```
[root@redhat certs]# hostname
redhat
[root@redhat certs]# cd /usr/share/ssl/certs
[root@redhat certs]# make slapd.pem
umask 77 ; \
PEM1=`/bin/mktemp /tmp/openssl.XXXXXX` ; \
PEM2=`/bin/mktemp /tmp/openssl.XXXXXX` ; \
/usr/bin/openssl req -newkey rsa:1024 -keyout $PEM1 -nodes -x509 -days 365 -
out $PEM2 ; \
cat $PEM1 > slapd.pem ; \
echo "" >> slapd.pem ; \
cat $PEM2 >> slapd.pem ; \
rm -f $PEM1 $PEM2
Using configuration from /usr/share/ssl/openssl.cnf
Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to '/tmp/openssl.NlHDoX'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:DE
State or Province Name (full name) [Berkshire]:Bavaria
Locality Name (eg, city) [Newbury]:Munich
Organization Name (eg, company) [My Company Ltd]:Example Labs
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:redhat
Email Address []:
[root@redhat certs]# openssl x509 -fingerprint -noout -in slapd.pem
MD5 Fingerprint=00:F2:33:CD:B5:80:10:63:A5:EB:2C:10:66:E0:E0:2D
[root@redhat certs]# chown ldap slapd.pem
[root@redhat certs]# chmod 600 slapd.pem
```

Configure OpenLDAP to use SSL

Now we need to configure the OpenLDAP server to actually use the just generated certificate. Open the `slapd.conf` file and locate the Transport Layer Security (TLS) section. Uncomment the TLS Paths for the keys (`TLSCertificateFile`) and the certificate (`TLSCertificateKeyFile`) and change the entries to point to the `slapd.pem` file (use absolute paths here):

```
#
# The next two lines allow use of TLS for connections using a dummy test
# certificate, but you should generate a proper certificate by changing to
# /usr/share/ssl/certs, running "make slapd.pem", and fixing permissions on
# slapd.pem so that the ldap user or group can read it.
TLSCertificateFile /usr/share/ssl/certs/slapd.pem
TLSCertificateKeyFile /usr/share/ssl/certs/slapd.pem
#
```

After restarting the OpenLDAP server SSL should be enabled and the server should listen on port 636. There is no need to manually configure the OpenLDAP server to listen on `ldaps` port. The start/stop script takes care of this automatically.

Configure VPN-1 to use LDAP over SSL

Open the LDAP Account Unit, choose the server you want to edit and select the encryption tab.

Activate the use of SSL and fetch the fingerprint of the LDAP server. Please verify that this is the same fingerprint you got when you created the certificate! Choose **Authentication** as minimum encryption strength. To use better encryption schemes consult the OpenLDAP documentation.

Restricting access

Right now we are using we are using the Directory Manager's Account to manage users, groups and templates with the Firewall GUI. The enforcement modules are using this account as well for LDAP queries when authenticating users. Although this works it may be a risk. The OpenLDAP Directory Manager has always full read/write access to everything. This may be too powerful for our needs:

In this section we will create two users: One is for the firewall enforcement module with read only rights for most of the attributes (some need write access). The second account will be for the Firewall administrator using the SmartDashboard GUI. This account will also have write access to attributes used by the firewall.

Creating accounts for the firewall and the administrator

We need to define the two accounts described above. `cn=Firewall,dc=example,dc=com` will be used by the firewall to query the LDAP server. `cn=Firewall Admin,ou=users,dc=example,dc=com` will be used by the firewall administrator to manage users, groups and templates on the LDAP server. Prepare an LDIF file like the one below (don't forget to create/change the password using `slappasswd`) and import it using `ldapadd` or the LDAP Browser.

```
dn: cn=Firewall,dc=example,dc=com
objectclass: person
objectclass: organizationalRole
cn: Firewall
```

```

sn: Firewall
description: This user is used by the firewall users to query the LDAP server
userpassword: {CRYPT}Z11HhzIDViC3M
#
dn: cn=Firewall Admin,ou=users,dc=example,dc=com
objectclass: person
objectclass: organizationalRole
cn: Firewall Admin
sn: Firewall Admin
description: This is by the firewall administrator
userpassword: {CRYPT}hfaUmdf6j2C6U

```

Setting access rights in slapd.conf

Now we need to set the access rights for the enforcement module and the administrator in slapd.conf. The enforcement module needs read access to all firewall attributes and write access to some password fields. The administrator needs full read/write access to all firewall attributes. Locate the access restriction section in slapd.conf and add the rules below:

```

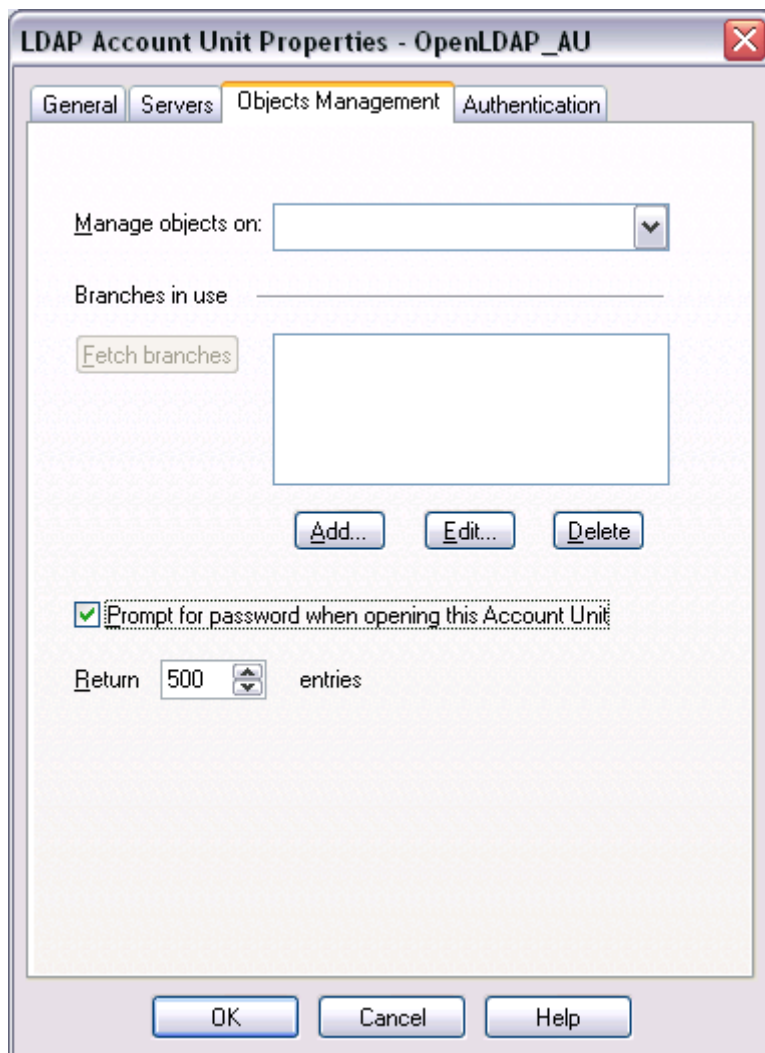
# Access control
#
# rootdn can always write!
#
access to dn=""
    by * read
access to attr=entry,cn,uid,description,mail,member
    by dn="cn=Firewall Admin,ou=users,dc=example,dc=com" write
    by * read
access to attr=userpassword
    by self write
    by dn="cn=Firewall,dc=example,dc=com" write
    by anonymous auth
access to attr=fwlpwdlastmod,fwlbadPwdCount,fwllastLoginFailure
    by dn="cn=Firewall,dc=example,dc=com" write
    by dn="cn=Firewall Admin,ou=users,dc=example,dc=com" write
    by * none
access to attr=fwlauth-method,fwlauth-server,fwlskey-number,fwlskey-seed, _
fwlskey-passwd,fwlskey-mdm,fwlexpiration-date,fwlhour-range-from, _
fwlhour-range-to,fwlday,fwllallowed-src,fwllallowed-dst,fwllallowed-vlan, _
fwlSR-keym,fwlsR-datam,fwlsR-mdm,fwlenc-fwz-expiration,fwlsr-auth-track, _
fwlgrouptemplate,fwlISAKMP-EncMethod,fwlISAKMP-AuthMethods, _
fwlISAKMP-HashMethods,fwlISAKMP-Transform,fwlISAKMP-DataIntegrityMethod, _
fwlISAKMP-SharedSecret,fwlISAKMP-DataEncMethod,fwlenc-methods, _
fwluserPwdPolicy,memberoftemplate,memberOf
    by dn="cn=Firewall,dc=example,dc=com" read
    by dn="cn=Firewall Admin,ou=users,dc=example,dc=com" write
    by * none
access to *
    by self write
    by users read
    by anonymous auth

```

Make sure no syntax error is found in slapd.conf by starting slapd manually in debug mode. If everything is fine you can (re)start the OpenLDAP server again.

Configure VPN-1 to use multiple LDAP accounts

VPN-1 is able to use different accounts for managing users, groups and templates from SmartDashboard and for actually fetching the LDAP server from the enforcement module. To activate this feature you should open the LDAP Account Unit properties and activate **Prompt for password when opening this Account Unit**.



Activating Prompt for password when opening this Account Unit

Now we need to tell the firewall which account should be used for the enforcement modules. Therefore we just need to replace the username and password in the LDAP Server Properties with our firewall account:

The next time you try to open an Account Unit you will be asked to enter your admin username and password:

Tips, Tricks and Troubleshooting

OpenLDAP 2.1.x and later

If you are using OpenLDAP 2.1.x and later you might run into problems with FireWall-1. VPN-1 uses LDAPv2 by default (the only exception is fetching the branches which is using LDAPv3). OpenLDAP 2.1.x does not support LDAPv2 by default anymore, just LDAPv3 (compile time option). If you have an OpenLDAP 2.1.x server you should add the following line to your slapd.conf file, which allows LDAPv2.

```
allow bind_v2
```

Refer to [OpenLDAP and LDAPv2](#) for further information.

IPv6

If you start slapd manually in debug mode you might get an error message complaining about a nonsupported Address family:

```
[root@redhat root]# /usr/sbin/slapd -d 768
daemon: socket() failed errno=97 (Address family not supported by protocol)
```

In most cases this message indicates that OpenLDAP was compiled with IPv6 support and is trying to open an IPv6 Socket additionally to the IPv4 socket although the Operating System does not support IPv6. Refer to [OpenLDAP and IPv6](#) for further information.

Migrating an existing OpenLDAP directory to be used by FireWall-1

If you already have a running OpenLDAP server you can add the schema file to slapd.conf while the server is running (see [Add the Check Point Schema definition](#)). There is no need to stop the OpenLDAP server while changing the slapd.conf file. Please verify the path of the VPN-1 schema file and make sure it is readable by the OpenLDAP server process. Afterwards you can restart the OpenLDAP server using `service ldap restart`.

Trying to fetch branches fails

Please check the following thing if it is not possible to fetch the branches in the Objects Management of the LDAP Account Unit:

Make sure the OpenLDAP server is running:

```
[root@redhat root]# service ldap status
slapd (pid 849 848 844) is running...
```

If you configured the server on the firewall side to use SSL please make sure the LDAP server is configured for SSL and is listening on port 636 (ldaps):

```
[root@redhat root]# netstat -l | grep -i -e 636 -e ldaps
tcp 0 0 *:ldaps:*:* LISTEN
```

If you configured the server on the firewall side to use SSL please make sure the LDAP server is configured for SSL and is listening on port 636 (ldaps):

Check whether the server supports LDAPv3. Although VPN-1 uses LDAPv2 for most operations fetching the branches is an LDAPv3 feature. If the LDAP server does not support LDAPv3 it is not possible to fetch the branches automatically you have to add them manually. Please refer to [OpenLDAP and LDAPv3](#) for further information.

How to check whether a LDAP license (Account Management Client) is installed

For using user databases on LDAP servers you'll need an Account Management Client license. The license is included in Enterprise and SmartCenter Pro. It is also available as an Add-on license feature; CPFW-AM-1-NG and CPFW-AM-U-NG. Use `cplic print` to check whether you have installed such a license:

```
[secureplatform]# cplic print
Host Expiration Features
172.16.1.2 21Feb2007 CPFW-AM-1-NG CK-0F1E2D3C4B5A
```

```
[secureplatform]# cplic print
Host Expiration Features
172.16.1.2 21Feb2007 CPFW-AM-U-NG CK-0F1E2D3C4B5A
```

If you do not have installed such a license it is still possible that the license feature is included in another license (e.g. SmartCenter Pro). To check whether your includes an appropriate feature you can use `cplic check`:

If you do not use an Enterprise Management you'll need to check for `am1`:

```
[secureplatform]# cplic check am1
cplic check 'ram1': license valid
```

If you use an Enterprise Management you'll need to check for `ram1`:

```
[secureplatform]# cplic check ram1
cplic check 'ram1': license valid
```

How to increase search speed when using dynamic groups

VPN-1 allows you to create dynamic LDAP Groups. These groups defined dynamically with LDAP search filters. This allows you to create groups, which are not just containing all users, users of a specified branch, or group but are also based on user properties (e.g. a mail address). Therefore you have the possibility to implicitly create user groups based on the users attributes (`mail=*muc.example.com`). You do not need to define these groups explicitly anymore (e.g. a group called `UsersInMunichLab`):

If you have a directory containing a lot of entries you might encounter delays when authenticating the users. This is especially true if you are using dynamic LDAP extensively. These delays are often caused by a non-optimal index on the LDAP server. If you are using a dynamic LDAP group which uses a search filter containing a `telephoneNumber` and the LDAP server uses no index for `telephoneNumber` the LDAP server is to check every entry. If the LDAP server contains "just" 100 entries this delay might not be noticeable; but if it contains tens of thousands of entries it will definitely be noticeable because the server has to check every single entry on its own. If `telephoneNumber` was indexed there is no difference between searching ten entries or tens of thousands of entries.

Therefore creating an index over attributes used in dynamic groups generally increases the speed of searches by orders of magnitudes. The default OpenLDAP configuration maintains an index for some often used attributes in `slapd.conf`:

```
# Indices to maintain
index objectClass,uid,uidNumber,gidNumber,memberUid eq
index cn,mail,surname,givenname eq,subinitial
```

If you use more attributes in your dynamic LDAP groups you should add them to `slapd.conf` as well.

```
# Indices to maintain
index objectClass,uid,uidNumber,gidNumber,memberUid eq
index cn,mail,surname,givenname eq,subinitial
index telephoneNumber pres,eq,approx,sub
```

After restarting the OpenLDAP server every *new* telephone number will create an index entry for presence, equality, approximated and substring search (please refer to the OpenLDAP manual for a detailed explanation of index types). Unfortunately just *new* telephone numbers will be indexed. To recreate the index for already created attributes you have to run `slapindex`. Please make sure the LDAP server is not running while reindexing the database.

```
[root@redhat root]# service ldap stop
[ OK ]
[root@redhat root]# slapindex -v
indexing id=00000011
indexing id=00000006
indexing id=00000004
indexing id=0000000d
indexing id=00000002
indexing id=0000000b
indexing id=00000009
indexing id=00000007
indexing id=00000010
indexing id=00000003
indexing id=0000000c
indexing id=00000001
indexing id=0000000a
indexing id=00000008
[root@redhat root]# service ldap start
Starting slapd:
```

Size limit exceeded

In some cases you might get an error message “size limit exceeded”.

This indicates that the LDAP server found more records than the firewall is willing to accept. You can increase the number of accepted records in the LDAP Account Unit Properties. Please be careful when setting this to a very high value. Especially when browsing from a very high point in the directory or when using the LDAP search tool you might encounter long delays during the transfer.

Note: Please note that LDAP servers have a size limit as well (`sizelimit` directive in `slapd.conf`. Refer to the OpenLDAP guide for further information). It’s important to know that an exceeded size limit on the server side cannot be recognized by the firewall. In these cases you won’t see all the entries the directory might have and get no error message.

Resources

OpenLDAP and LDAPv2

ldap_bind: Protocol error

http://www.openldap.org/faq/index.cgi?highlightWords=bind_v2&file=693

How to configure OpenLDAP 2.1.x slapd for my old (LDAPv2) clients to keep working?

http://www.openldap.org/faq/index.cgi?highlightWords=bind_v2&file=822

OpenLDAP and LDAPv3

What to do when "Fetch branches" button in the Account Unit Properties window does not work?

<http://secureknowledge.checkpoint.com/SecureKnowledge/viewSolutionDocument.do?id=sk12286>

What LDAPv3 features/extensions are supported by the OpenLDAP server?

<http://www.openldap.org/faq/index.cgi?highlightWords=ldapv3&file=645>

What LDAPv3 features/extensions are NOT supported by the OpenLDAP server?

<http://www.openldap.org/faq/index.cgi?highlightWords=ldapv3&file=649>

ObjectClass Violations

Common errors ldapadd/modify

<http://www.openldap.org/faq/data/cache/650.html>

OpenLDAP and IPv6

daemon: socket() failed errno=97 (Address family not supported)

<http://www.openldap.org/faq/index.cgi?highlightWords=address%20family%20not%20supported&file=652>

Linux IPv6 HOWTO

http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Linux+IPv6-HOWTO.html

OpenLDAP and LDAP over SSL

OpenLDAP, OpenSSL, SASL and KerberosV HOWTO

<http://www.bayour.com/LDAPv3-HOWTO.html>

VPN-1 schema extension

What LDAP schema changes were implemented in VPN-1NG?

<http://secureknowledge.checkpoint.com/SecureKnowledge/viewSolutionDocument.do?id=sk13510>

Appendix

Netscape LDIF Schema to OpenLDAP schema conversion

The schema extension for the Netscape Directory Server is delivered in LDIF format. The LDIF file can be found at \$FWDIR/lib/ldap/schema.ldif on a VPN-1 machine. With a few search and replace operations it is possible to translate this into a format which can be used by OpenLDAP. This is a simple shell/sed script which does the conversion.

Note: Be sure to add AUXILIARY to the fw1person objectClass definition.

```
#!/bin/sh
# convertSchema - (c) 2003 by Udo Schneider <udos@checkpoint.com>
#
# This script converts the Check Point schema extensions for the
# Netscape directory server to a format which can be used by OpenLDAP
#
# Usage:
# This script reads the Netscape LDIF file and writes the OpenLDAP
# schema file to standard output.
# You can use it like this:
#
# convertSchema $FWDIR/lib/ldap/schema.ldif > $FWDIR/lib/ldap/fw1ng.schema
#
grep -e objectclasses: -e attributetypes: $1 \
| sed -e 's/attributetypes:\n\n/nattributetype/g' \
-e 's/objectclasses:\n\n/nobjectclass /g' \
-e 's/NAME/\n NAME/g' \
-e 's/SYNTAX/\n SYNTAX/g' \
-e 's/SUP/\n SUP/g' \
-e 's/MUST/\n MUST/g' \
-e 's/MAY/\n MAY/g'
```

Check Point schema extension file for OpenLDAP (fw1ng.schema)

SmartDirectory (LDAP) Schema

Proprietary OID

Each of the proprietary object classes and attributes (all of which begin with “fw1”) has a proprietary Object Identifier (OID), listed below.

Objectclasses

TABLE 8-2 Object Class OIDs

object class	OID
fwltemplate	1.3.114.7.3.2.0.1
fwlperson	1.3.114.7.3.2.0.2

The OIDs for the proprietary attributes begin with the prefix (“1.3.114.7.4.2.0.X”). Only the value of “X” is different for each attribute. See “Attributes” for the value of “X”. The syntax for each is IA5 String, 1.3.6.1.4.1.1466.115.121.1.26.

Attributes

attribute	OID	fwlperson	fwltemplate	defaults
fwlauth-method	.1	yes	yes	undefined
fwlauth-server	.2	yes	yes	
fwlpasswdlastmod	.3	yes	yes	no value
fwlexpiration-date	.8	yes	yes	no value
fwlhour-range-from	.9	yes	yes	“00:00”
fwlhour-range-to	.10	yes	yes	“23:59”
fwlweekday	.11	yes	yes	all days of the week
fwlallowed-src	.12	yes	yes	no value
fwlallowed-dst	.13	yes	yes	no value
fwlallowed-vlan	.14	yes	yes	no value
fwlSR-keym	.15	yes	yes	Any
fwlSR-datam	.16	yes	yes	Any
fwlSR-mdm	.17	yes	yes	none
fwlenc-fwz-expiration	.18	yes	yes	
fwlsr-auth-track	.19	yes	yes	none
fwlgrouptemplate	.20	yes	yes	false
fwlISAKMP-EncMethod	.21	yes	yes	DES, 3DES
fwlISAKMP-AuthMethods	.22	yes	yes	signatures
fwlISAKMP-HashMethods	.23	yes	yes	MD5, SHA1
fwlISAKMP-Transform	.24	yes	yes	ESP
fwlISAKMP-DataIntegrityMethod	.25	yes	yes	SHA1
fwlISAKMP-SharedSecret	.26	yes	yes	
fwlISAKMP-DataEncMethod	.27	yes	yes	DES
fwlenc-methods	.28	yes	yes	FWZ
fwluserPwdPolicy	.29	yes	yes	
fwlbadPwdCount	.30	yes		
fwllastLoginFailure	.31	yes		
memberoftemplate	.32	yes		
memberof	.33	yes	yes	

cn

The entry’s name. This is also referred to as “Common Name”. For users this can be different from the uid attribute, the name used to login to the VPN-1 Pro module. This attribute is also used to build the SmartDirectory (LDAP) entry’s distinguished name, that is, it is the RDN of the DN.

uid

The user's login name, that is, the name used to login to the VPN-1 Pro module. This attribute is passed to the external authentication system in all authentication methods except for "Internal Password", and must be defined for all these authentication schemes.

The login name is used by VPN-1 Pro to search the SmartDirectory (LDAP) server(s). For this reason, each user entry should have its own unique uid value. It is also possible to login to the VPN-1 Pro module using the full DN. The DN can be used when there is an ambiguity with this attribute or in "Internal Password" when this attribute may be missing. The DN can also be used when the same user (with the same uid) is defined in more than one Account Unit on different SmartDirectory (LDAP) servers.

member

An entry can have zero or more values for this attribute.

In a template: The DN of user entries using this template. DNs that are not users (object classes that are not one of: "person", "organizationalPerson", "inetOrgPerson" or "fwlperson") are ignored.

In a group: The DN of user.

userPassword

Must be given if the authentication method (fwlauth-method) is "Internal Password". The value can be hashed using "crypt". In this case the syntax of this attribute is: "{crypt}xyyyyyyyyyyy" where "xx" is the "salt" and "yyyyyyyyyy" is the hashed password. It is possible (but not recommended) to store the password without hashing. However, if hashing is specified in the SmartDirectory (LDAP) server, you should not specify hashing here, in order to prevent the password from being hashed twice. You should also use SSL in this case, to prevent sending an unencrypted password. The VPN-1 Pro module never reads this attribute, though it does write it. Instead, the SmartDirectory (LDAP) bind operation is used to verify a password.

```

attributetype ( 1.3.114.7.4.2.0.1
    NAME 'fwlauth-method'
    SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.2
    NAME 'fwlauth-server'
    SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.3
    NAME 'fwlpwdlastmod'
    SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.4
    NAME 'fwlskey-number'
    SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.5
    NAME 'fwlskey-seed'
    SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.6
    NAME 'fwlskey-passwd'
    SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.7
    NAME 'fwlskey-mdm'

```

```
SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.8
  NAME 'fwlexpiration-date'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.9
  NAME 'fwlhour-range-from'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.10
  NAME 'fwlhour-range-to'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.11
  NAME 'fwlday'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.12
  NAME 'fwlallowed-src'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.13
  NAME 'fwlallowed-dst'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.14
  NAME 'fwlallowed-vlan'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.15
  NAME 'fwlSR-keym'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.16
  NAME 'fwlSR-datam'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.17
  NAME 'fwlSR-mdm'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.18
  NAME 'fwlenc-fwz-expiration'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.19
  NAME 'fwlsr-auth-track'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.20
  NAME 'fwlgrouptemplate'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.21
  NAME 'fwlISAKMP-EncMethod'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.22
  NAME 'fwlISAKMP-AuthMethods'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )
attributetype ( 1.3.114.7.4.2.0.23
  NAME 'fwlISAKMP-HashMethods'
```

```
SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.24
  NAME 'fwlISAKMP-Transform'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.25
  NAME 'fwlISAKMP-DataIntegrityMethod'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.26
  NAME 'fwlISAKMP-SharedSecret'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.27
  NAME 'fwlISAKMP-DataEncMethod'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.28
  NAME 'fwlenc-methods'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.29
  NAME 'fwluserPwdPolicy'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.30
  NAME 'fwlbadPwdCount'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.31
  NAME 'fwllastLoginFailure'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.32
  NAME 'memberoftemplate'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

attributetype ( 1.3.114.7.4.2.0.33
  NAME 'memberOf'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.26' )

objectclass ( 1.3.114.7.3.2.0.1
  NAME 'fwltemplate'
  SUP 'top'
  MUST ( cn )
  MAY ( member $ description $ fwlauth-method $ fwlauth-server $
fwlpwdlastmod $ fwlskey-number $ fwlskey-seed $ fwlskey-passwd $ fwlskey-mdm $
fwlexpiration-date $ fwlhour-range-from $ fwlhour-range-to $ fwlday $ fwllallowed-
src $ fwllallowed-dst $ fwllallowed-vlan $ fwlsr-keym $ fwlsr-datam $ fwlsr-mdm $
fwlenc-fwz-expiration $ fwlsr-auth-track $ fwlgrouptemplate $ fwlISAKMP-EncMethod
$ fwlISAKMP-AuthMethods $ fwlISAKMP-HashMethods $ fwlISAKMP-Transform $ fwlISAKMP-
DataIntegrityMethod $ fwlISAKMP-SharedSecret $ fwlISAKMP-DataEncMethod $ fwlenc-
methods $ fwluserPwdPolicy $ memberOf) )

objectclass ( 1.3.114.7.3.2.0.2
  NAME 'fwlperson'
  SUP top AUXILIARY
  MUST ( cn $ sn )
  MAY ( description $ userpassword $ mail $ uid $ fwlauth-method $ fwlauth-
server $ fwlpwdlastmod $ fwlskey-number $ fwlskey-seed $ fwlskey-passwd $ fwlskey-
mdm $ fwlexpiration-date $ fwlhour-range-from $ fwlhour-range-to $ fwlday $
fwllallowed-src $ fwllallowed-dst $ fwllallowed-vlan $ fwlsr-keym $ fwlsr-datam $
fwlsr-mdm $ fwlenc-fwz-expiration $ fwlsr-auth-track $ fwlgrouptemplate $
```

```
fwlISAKMP-EncMethod $ fwlISAKMP-AuthMethods $ fwlISAKMP-HashMethods $ fwlISAKMP-  
Transform $ fwlISAKMP-DataIntegrityMethod $ fwlISAKMP-SharedSecret $ fwlISAKMP-  
DataEncMethod $ fwlenc-methods $ fwluserPwdPolicy $ fwlbadPwdCount $  
fwllastLoginFailure $ memberoftemplate $ memberOf) )
```